AD A127030

Research Note 81-13

APPLICATION OF RULE-BASED COMPUTER MODELS TO THE EVALUATION OF COMBAT TRAINING: A FEASIBILITY STUDY

Efraim Shaket, Joseph Saleh, and Amos Freedy
Perceptronics, Inc.

ARI FIELD UNIT AT PRESIDIO OF MONTEREY, CALIFORNIA

U. S. Army
Research Institute for the Behavioral and Social Sciences

July 1981

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Research Note 81-13	2. GOVT ACCESSION NO. AD-A127 050	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Application of Rule-Based Computer Models to the Evaluation of Combat Training: A Feasibility Study		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report August 1979-July 1980
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Efraim Shaket, Joseph Saleh, Amos Freedy		8. CONTRACT OR GRANT NUMBER(s) MDA903-78-C-2014
9. PERFORMING ORGANIZATION NAME AND ADDRESS Perceptronics, Inc. 6271 Variel Avenue Woodland Hills, California		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q162722A791
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Research Institute for the Behavioral & Social Sciences 5001 Eisenhower Avenue, Alexandria, VA 22333		12. REPORT DATE
		13. NUMBER OF PAGES 90
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This work was conducted under the technical monitorship of Dr. James Banks, ARI Field Unit, Presidio of Monterey, California.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Training System Evaluation
Rule-Based Models
Interactive Programs
Computer Program Assessment

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the approach, product, and results of an exploratory effort in employing computer rule-based techniques and adaptive programming technology to the modeling of small unit, tactical engagements. The model was to be used as a training evaluation aid to improve exercise evaluation in training systems, such as REALTRAIN, SCOPE and MILES. The techniques are expected to lead to greater training effectiveness by providing the training officer with a means for identifying lacking skills and incorrect tactical behaviors in the exercising unit.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

The focus of the work was on the adaptation of a rule-based, event-driven model to the representation of tactical engagements. The model describes a military mission as a hierarchy of tasks performed by a unit and its components. The tasks are connected by production rules-conditional events that cause transition to new tasks. This model, when represented explicitly in a computer, provides the framework for the implementation of an interactive computer program for evaluation of tactical tasks performed during a field exercise. The explicit model allows the computer program to compare directly the preferred solution of the exercise to what was actually performed in the exercise, and to identify the significant intermediate steps that caused eventual success or failure.

A demonstration package was implemented as part of this effort. It includes interactive programs, written in the PASCAL programming language, which handle man-machine communication. The demonstration simulates a typical interaction between a training officer performing a post-exercise evaluation of a tank platoon. The tactical knowledge-base is limited in scope at this stage to the "Bounding Overwatch" maneuver during the "Move to Contact" phase of a Hasty Attack. The program's responses are derived by comparing the internal description of this mission with the user's input; and, if he needs assistance, by prompting him with information available from the tactical knowledge base.

EXECUTIVE SUMMARY

This report describes the results of a study exploring the application of two modern computer techniques, knowledge-based modeling and adaptive programming technology to the analysis of small-unit tactical engagements. The underlying purpose of the study is to improve exercise evaluation in complex and realistic tactical training systems, such as the recently introduced MILES. The techniques are expected to lead to greater training effectiveness by providing trainers with computer aids which, on the basis of real-time exercise data, identify incorrect tactical behaviors in the exercising units, and suggest required training directions.

Such computer aiding requires software that can describe in detail a variety of tactical situations, and can facilitate the related representation of performance data. In essence, the software must have the ability to combine training data from diverse sources into an integrated model of the simulated engagement.

The focus of the present work was on examining the feasibility of a rule-based, event-driven, computer model for the representation of small-unit combat engagements and for subsequent performance evaluation. The rule-based, event-driven approach was selected over other types of decision process models for several reasons: (1) it can efficiently describe numerous tactical relationships; (2) it can be readily expanded; and (3) it is inherently structured to respond to diverse external occurrences.

The fundamental principle employed in model development was the conception of the battlefield as a large collection of units, each performing its mission by taking initiatives and responding to other units and ele-

ments. This concept suggests that a useful description of tactical behavior has to contain the standard, expected way of performing a mission; but it must also include the potential responses to many types of external events caused by actions of the opponent force or by other occurrences in the tactical environment. The totality of these tactical and behavioral descriptions is termed the "knowledge base" of the system. The constituent "rules" of the knowledge base, referred to above, are of the type "If Event A happens, then perform (initiate) Action X." Although the rules themselves are simple, there are many of them, and, with the power of a computer, they form a complete model of a complex situation.

The feasibility of this modeling approach in the area of tactical training was demonstrated by first creating a model schema and a knowledge base for the "Move to Contact" phase of a "Hasty Attack" by a tank platoon, and then using this knowledge base to develop an interactive program which simulates a post-exercise evaluation. The demonstration program was written in PASCAL, and implemented on a PDP 11/2 microcomputer system. From its "knowledge" of the move to contact maneuver, the program can analyze a hypothetical exercise to: (1) isolate key events in the actual scenario, highlighting failures to initiate actions or to respond properly to external events; (2) summarize a unit's utilization of resources; and (3) identify component skills that did not achieve acceptable performance levels.

The study described in this report was a pilot effort, intended only to establish feasibility of approach. It represents a beginning point for a wide range of future computer-based aiding systems. These may include systems which contain much larger knowledge bases, systems which support automated exercise data collection, and systems which extend the training evaluation in the direction of automated inference. Ultimately,

such systems will provide the user with explicit diagnostic information on the relationships between tactical activities and combat performance, so that specific remedial training can be prescribed immediately after a simulated engagement is completed.

APPLICATION OF RULE-BASED COMPUTER MODELS TO THE EVALUATION OF COMBAT TRAINING: A FEASIBILITY STUDY

CONTENTS

CONTENTS (Cont'd)

1. INTRODUCTION

1.1 Overview

This report describes the products and results of an exploratory effort in employing adaptive programming technology for tactical models in order to provide real time aids for the improvement of military exercise evaluation in tactical engagement simulation systems, such as MILES. These techniques can lead to greater training effectiveness by providing the training officer with a means for integrating and analyzing the tremendous amount of data that is generated by such systems at high speeds by and assisting him in identifying critical teaching points.

The result of this effort provides an initial technological basis for automated After-Action Evaluation (AAE) in the "experimental training environment" which is included in the National Training Center functions. Typical to this training environment is the large amount of performance data which is collected from concentrated exercises and must be processed in a short time. Good tactics and bad tactics must be identified and corrected by establishing teaching points regarding specific decisions and actions. In order to fulfill the evaluation requirements of NTC, real time computer evaluation aids are essential. The feasibility of such evaluation aids and their effectiveness have been investigated in this report.

The report also provides a presentation of how these techniques impact the training cycle, especially in the evaluation phase. This includes a discussion of the specific problems with the current evaluation systems and identification of points of potential improvement, as well as a projected evaluation system that uses the computer aided techniques. A discussion of future phases of the effort that are required for a move

towards an operational product in an environment such as the National Training Center is also presented.

1.2 Scope of Effort

This first effort focused on the adaption of the rule-based event driven abstract model to the modelling of tactical engagements. The model describes a military mission as a hierarchy of tasks performed by a unit and its components. The tasks are connected by production rules--conditional events that cause transitions to new tasks. This mode, when represented explicitly in a computer, provides the framework for the implementation of an interactive computer program for evaluation of tactical tasks performed during a field exercise. The explicit model allows the computer program to compare directly the preferred solution of the exercise to what was actually performed in the exercise and identify the significant intermediate steps that caused eventual success or failure.

A demonstration package was implemented as part of this effort. The implementation of the system tactical knowledge base was limited in scope to the "Bounding Overwatch" maneuver during the "Move to Contact" phase of a Hasty Attack. The demonstration was directed toward the elementary function of interactive information elicitation from the evaluation officer and generation of a summary report.

This effort constitutes a feasibility study not only by demonstrating that an effective evaluation aid can in principle be constructed, but by selecting a proper model and actually constructing a working tool. Naturally, at this early stage of the development, only a limited example could be used in demonstrating the immediate evaluation aiding capabilities of the model. Other feasibility issues were resolved in the APT report (Shaket, 1978), showing that the modeling techniques are avail-

able and were applied successfully in other applications; the compatible programming technology has been tested and refined, and hardware power is available now and more will be in the next few years.

1.3 Rationale

1.3.1 The Military Problem. The improvement of tactical training has become one of the highest Army priorities. To improve training effectiveness, the ARTEP (Army Training and Evaluation Program) was launched with an increased emphasis on the development of performance oriented training and evaluating methods. Several training simulation systems were developed by ARI to overcome ARTEP's major weakness--the previous lack of an objective way to determine terminal mission outcome. These simulation systems are SCOPES, REALTRAIN, and now Multiple Integrated Laser Engagement Systems (MILES), which is in the initial introduction phase. They provide the commander with the capability to conduct two-sided, free-play tactical exercises with credible casuality assessment, weapon signature effects and high realism.

Because of the large amount of information that will be available to the training officer when these and future simulation systems are used, he will need help in the analysis and evaluation of the data. The T&EO documents that are being developed provide performance standards and detailed performance measures for various tactical maneuvers. But, being based on paper documents, they suffer from several major drawbacks. First, they are voluminous and inaccessible, and they require the training officer to manually search through the information to find what is relevant to his exercise.

In order to be of value to its users, the T&EO must be stated in general terms, and thus its applicability to any particular case diminishes. Also, what is relevant to any particular scenario is embedded without distinction between many facts and causes that are only relevant to other scenarios. The "conditions" column of the T&EO is provided to indicate what is relevant in a given case and what should be ignored. In fact, the sequential linear layout of the T&EOs (a characteristic of any book) makes reading it quite awkward because of all the skipping that has to be done. The ARTEP system in general, and the T&EO in particular, leaves much to be desired in the area of active assistance to the training officer in his analysis, planning, and evaluation of the training task he is faced with.

This project, therefore, was designed to investigate the use of a computer system (incorporating a rule-based model and adaptive programming) to provide active assistance to the training officer. It produced a demonstration program that can interactively assist him in establishing the sequence of events that occurred in the exercise, suggesting the relevant performance measures and standards of performance, and finally, producing a summary of the main events, performance levels obtained, and key tactical failures.

The advantage of a computer-based system stems from (1) its information processing capability, (2) its ability to store large volumes of information while accessing it selectively and almost instantly, (3) its capability to adapt its response to the changing situation, and (4) its responsiveness (within the limits of its design) to the demands of the user. Comparing these capabilities to the T&EO, it can be said that a computer-based evalution system can present to the user a tailor-made description of the task, consistent with the prevailing tactical situation and environment; user analytic preference; and the particular se-

quence of events that transpired during a particular exercise run. At the same time, it can serve as a valuable discrete tutorial tool that gives definitions of tasks, expected events, expected standards of performance, etc., but only when requested by the user and always on hand. The essence of a computer-based model is that it is explicitly stated, and the computer program itself can manipulate the internal description and present to the user only the relevant information in the exact format that he wishes to see it.

1.3.2 Relevance to NTC. A very relevant application of real time evaluation aids is in heavily instrumented engagement simulation systems, such as the National Training Center, Ft. Irwin, California. The mission of NTC is to provide intensive battalion training in a realistic combined arms combat environment. NTC will utilize MILES (Multiple Integrated Laser Engagement System) to assess realistic kill and hit and measure suppressive fire effects under approximate tactical conditions.

This information, along with range measurement data is relayed through data links to a central processor and a core instrumentation system, where it is integrated with obsearvers' data and formated and displayed for training evalution for After-Action review. (Agnew, 1980.)

Evaluators have at their disposal many valuable tools which can be used to analyze the combat activities and evolve teaching points for the After-Action review. Some of these tools include the capability to instantaneously select different map scales, zoom levels, and areas; the ability to display terrain features, such as grid line, contour lines, roads, trails and rivers, and a mobility index background; and the ability to defray control area, control boundaries and lines and control points. We can also display different echelons of the player units, run the exercise in a fast time mode until we observe a critical event,

which can then be tagged and instantly replayed at a different zoom level, with the individual weapons represented. The system also provides summary tables during real time, so we observe such things as killer target scoreboards and distribution of fires. The graphic displays of information also assist us in editing the tactical field video tapes and voice recordings into the After-Action review. The total system enables us to reconstruct the battle for the unit command, assist him in determining why things happened as they did and make strong teaching points.

In order to take full advantage of the large amount of data and provide meaningful evaluation at a fast time response, real time computer aids for the evalution are essential. Such computer aids require models that can describe the tactical situations and events and facilitate representation of the data in computer software. In particular they have the capability to integrate data from diverse sources into an integrated engagement model, identify key tactical decisions and events, and provide an aid to determine critical teaching points. The ultimate objective of real time computer aids is to provide inferential information which explains the response behind certain actions.

1.4 Approach

A tactical battlefield can be thought of as a large collection of units performing their missions, taking initiative, and responding to events and actions by other units. This oversimplified statement suggests that a useful description of tactical behavior must be event-driven. It has to contain the normal, expected way of performing a mission; but it should include also the required resonse to many possible external events caused by the Opponent Force (OPFOR) or the tactical environment. In a particular engagement, the general mission of all the units involved, and the particular sequence of events that occurred, "unfold"

into a unique scenario. This project has developed a computer based model and a simple implementation that capture this essential characteristic of the military environment.

The production-rules model describes a tactical mission as a hierarchy of connected tasks. The tasks are the components of the mission, and they in turn are represented by their component subtasks down to the level of detail needed or useful. The tasks are connected by arcs, representing events. These events are the possible reasons or causes of terminating the task the arc starts from. The arc points to the task that should be started when and if its event occurs. The collection of tasks and events thus describes a tactical mission, all the probable events that may occur in the various phases, and the expected responses by the unit. A particular exercise or a real engagement will trace a path through this network, starting from some initial task and ending at some terminal task. Each task description contains a list of the actions that have to be taken to accomplish the task, some standards of performance, and detailed performance measures if applicble. Together, these descriptions make up the tactical knowledge base which a properly designed computer program can access and manipulate effectively.

The interactive computer program conducts a dialog with the training officer who performs an exercise evaluation. It asks him to provide the tasks performed by the engaging units, the event that occurred, and the unit's responses to them. It then compares these inputs to the tactical knowledge base it can access directly, and comes up with the following types of evaluations:

(1) Which phases of the mission the unit failed to perform altogether.

(2) Which events (actions by OPFOR) it failed to detect, respond to or responded inappropriately.

(3) Which procedures were not carried out appropriately.

(4) Which, if any, was the key point (any of the above) that contributed to mission failure.

(5) Which resources were depleted, misused or misappropriated.

(6) Which are the skills that demonstrably did not achieve acceptable levels of performance.

These evaluations can be directly derived from the model and the input provided by the user. They can be the first phase of a much more sophisticated system that can perform inference and identify indirect causes to failure and generalized evaluation of tactical skills.

The advantages of such an interactive computer-based evaluation system are many: (1) it will reduce the training specialization required of an officer to perform effective training within training systems such as REALTRAIN or MILES, (2) it will improve training effectiveness by helping identify reasons for failure to accomplish a mission and the specific skills that a unit demonstrably lacks, and (3) it will improve training efficiency by helping the officer plan training scenarios that directly address the skills that need further training. Furthermore, such a computer-based system can facilitate technology transfer of MILES to the field units by replacing a large set of paper manuals with a computer that can access and present to the user only information relevant to his particular training objectives, interactively.

The technique used in the model implementation is the production rule-based system. This technique was developed under the generic area of Artificial Intelligence and is now ripe for applications. Several successful applications already have been demonstrated in diverse and complex areas such as medical diagnosis, chemical modeling, and mineral exploration.

The PASCAL programming language was selected for the implementation part of this project because it facilitates fast construction of complex and large programs with less developer induced errors. Furthermore, it is capable of handling the complex data structures that were involved in the tactical knowledge base.

To adapt production rule techniques to the specific needs of military exercise evaluation in tactical engagement simulation systems, the project started off with an analysis of the requirements of a tactical evaluation aiding tool. As presented in detail in Chapter 2, the desired tool should adapt dynamically to the changing tactical situation and the evaluator's specific current requirements. It should selectively present him only with data that is relevant to the events being considered and actively participate in the evaluation process by prompting the user for expected events and conditions. In terms of functionality, the system has to be portable so that it can be used in the field modular so that different capabilities can be provided for users at different levels and integrated by providing a global solution to all the phases of the evaluation process. The model and software must also be incrementally modifiable, so that as new tactics evolve and new evaluation methods are developed, they can be incorporated into the working system without too much effort and redesign.

The feasibility of the Adaptive Programming Technology has been analyzed in previous efforts (Shaket, 1978; Alperovitch and Shaket, 1980; Shortliffe, 1976). Essentially, it rests on the availability of models, techniques, technology, and track record. Applicable models have been developed over the past twenty years under the generic name Artificial Intelligence. Programming techniques have been refined at MIT, STANFORD, and other AI centers culminating in languages like LISP and operating systems like INTERLISP (INTERLISP reference manual XEROX, 1974). Computer technology continues to run forward doubling in performance per chip every year easily outpacing progress in models and programming. Long term (10 years) predictions indicate that the computing power of the largest computers of today may be placed on a single large chip. In the last few years this combined technology has matured into several highly successful planning, decision and diagnostic aids especially in medical diagnosis (Shortliffe, 1976), mineral exploration (Hart and Duda, 1977) and molecular analysis (Buchanan, 1976). These and other APT based systems have demonstrated levels of performance and sophistication commonly associated with experts in their respective domain. Having demonstrated feasibility, one additional point that has been learned must be stressed. It is that the knowledge base in each of these applications is highly specific and limited in scope to that particular domain. The construction and refinement of a tactical knowledge base is a major effort that can only be started in this project and generate directions for future developments. These are therefore the objectives of the current effort.

Several theoretical models have been evaluated in a previous effort (May, Shaket and Leal, 1979) and in this project. Among others they include: the elicited probability approach (Steeb et al, 1973), adaptive decision modeling approach (Freedy et al, 1976), the heuristic search approach (Nilsson, 1971), and the production rules approach (pattern

directed inference). The dynamic, event-driven nature of the tactical engagement environment made the production rules approach the best choice (See discussion in Chapter 3.). To show the feasibility of the model and the approach, this project concentrated on mapping the general production rules model into a specific tactical evaluation task, identification of failure to perform intermediate task, improper responses to tactical events, and aggregation of evaluative data provided by the user.

The direct representation of tactical tasks and events provided by the model and its event driven character lends itself naturally to a tactical evaluation tool. Given the events that actually happened in the ongoing exercise, the computer can access its tactical knowledge base and compare the actions taken by the training unit with the proper response to these events. In the future NTC enviornment, the data collection will be at least partially automated, but in this small demonstration program the evaluator has to provide the data himself. It must be noted, however, that even if most of the raw data were to be collected automatically its interpretation and evaluation would have to be done by experts. The same physical move of a tank down a hill may be considered correct or incorrect depending on subtle differences in time, location of the OPFOR or relation to other forces.

Another key benefit of an event driven model is that it can present to the user instantly all the data relevant to the actual events that happened and hide large amounts of data related to event that might have happened but did not. Compare this to a document based ARTEP, where all the information about possible events must be explicitely stated sequentially, and the user must sift through the mounds of documents to get to the few tasks that are actually relevant.

1.5 Current System Implementation

A demonstration package was implemented as part of this effort. It included an interactive program, written in the PASCAL programming language. The program demonstrates a typical interaction between the knowledge base and a training officer performing a post-exercise evaluation of a tank platoon. The system tactical knowledge-base is limited in scope at this stage to the "Bounding Overwatch" maneuver during the "Move to Contact" phase of a Hasty Attack. All the examples and discussion in this report refer to this set of tactical tasks. All the program's responses are derived by comparing the internal description of this mission with the user's input; and if he needs assistance, by prompting him with information learned from the tactical knowledge base. Future systems will naturally have a much larger tactical knowledge-base and a more elaborate set of evaluation, comparison, and deduction mechanisms.

The hardware used in this system is a portable microcomputer. The computer was actually transported from LA to Monterey in two boxes and set up in short order. The key features of this computer are:

(1) Digital Equipment Corp. LSI 11/2 16 Bit Microprocessor.

(2) 64000 Bytes of RAM memory.

(3) 4 Serial I/O communication channels.

(4) Dual floppy disks with 1.2 Million Bytes of backup memory.

(5) 24x80 full CRT display.

(6) UCSD PASCAL(T) operating system.

The program itself took more than 1000 lines of PASCAL code and occupies about 25000 Bytes of memory. Its reaction time for most requests by the user is not more than 2 seconds. Even with its limited scope, the demonstration program itself can be used to identify key events, failure to perform proper response, and simple summary of skill attainment, together with interactive presentation of relevant ARTEP-type data.

1.6 Future Phases

The effort described in this report was a pilot study, intended to establish feasibility of approach. It is a beginning phase for a wide range of possible future computer-based systems. We propose the directions of additional efforts along the following three dimensions:

(1) Developing and enlarging the scope of the military knowledge base up to combined arms exercises.

(2) Developing the assistance capabilities in training evaluation.

(3) Expanding the software and computer capabilities.

In the first dimension, we can naturally expand the number, size, and complexity of the missions covered. We can take into account different terrain, weather, and enemy compositions. We can cover the missions of larger units up to and including the division level, and we can expand to other types of training units: infantry, artillery, air-ground support, and even naval operations. In fact, a similar model was used

effectively in a previous effort which simulated a responsive knowledgeable opponent for a trainee submarine commander.

The second dimension covers expanded capabilities in aiding a tactical training evaluation. The system can identify tasks and responses that lead to a dead end or to an unacceptable outcome. It may perform higher-level summarization of the performed tactical skills, classifying them by general tactical skills, or it may be able even to explain events that are peculiar to a specific exercised run and do not carry over to other missions or tasks. Finally, it may be made to perform logical inference to deduce causes of success and failure from recognized incomplete or unacceptable performance measures.

The event-driven model can be used also in aiding other functions of the training officer. Following the training cycle described in Chapter 2 (also in ARTEP T&EO manual...), the system can be built to assist in planning an effective training exercise, in planning the data collection arrangement, and in the initial analysis that is done to identify the unit's training needs in the first place.

The third dimension covers improvement in the software and computer systems to provide more convenience and ease of use. This may include more natural-language-like interaction, entering of data by multiple users, where each inputs the events and tasks of some subunits, collection and reduction of data from many remote data transmitters, etc. The software improvements will, naturally, have to grow hand to hand with the expansions in scope along the other dimensions.

1.7 Report Organization

This report starts off with an analysis of the tactical training process (Chapter 2) presenting it as a cyclical process. It then identifies problems with the current evaluation aids and sets a list of requirements that an ideal evaluation aid should have. Chapter 3 gives a short description of several possible models, settling on the production rules model as the most promising for a tactical training evaluation tool. It then goes on to describe the principles of this model and how it can be applied to represent a tactical engagement. Chapter 4 presents the tactical schema of part of the Hasty Attack mission of a tank platoon. This is the mission chosen for the current demonstration program. Chapter 5 includes a description of the demonstration package itself, the system process, a sample dialog, and an example of the evaluation summary report that future systems could produce. Finally, Chapter 6 discusses future systems applications of the tactical model introduced in this effort. It classifies the possible extentions along three general dimensions:

(1) Expanding the evaluation capabilities.

(2) Expanding the military scope.

(3) Expanding the software and system scope.

The appendix includes a listing of the PASCAL program of the demonstration package.

2. PROBLEM ANALYSIS

2.1 Overview

This section is a summary of the problem analysis performed during this effort. It starts with a description of the training process as a training cycle in which (1) analysis, (2) planning, (3) evaluation planning, (4) execution, and (5) evaluation follow each other in order and then lead to the next cycle. Section 2.3 presents some of the past and present solutions that were applied to the improvement of this process. Section 2.4 concentrates on the evaluaticn process, describes in general terms the ARTEP system, and identifies some of its drawbacks. Section 2.5 shows desirable points of improvements in the evaluation process, and Section 2.6 concludes with a list of capabilities that an improved evaluation system should have and a description of how each capability will improve the evaluation.

Note that this pilot effort concentrated on the tactical evaluation process because it was judged most amenable to assistance by a computer-based aiding device. Other parts of the training process also can be improved with a properly developed aid, but that would constitute a separate future effort.

2.2 A Training Cycle

The field training process, when conducted around a realistic simulation system such as MILES, can be considered a cyclic process. The objective of this process is to improve the tactical performance of the training unit through repeated exercising of maneuvers which contain tactical skills that the unit is observed to be lacking. After each exercise, there is an evaluation phase, in which the training officer tries to

determine what tactical skills have been satisfactorily demonstrated and what require further training. Breaking this cycle into finer details, we can identify the following phases:

(1) Analysis: Identify the missions to be trained for and the collection of skills required for a satisfactory performance level.

(2) Planning: Develop an efficient training scenario that, while spending minimal resources, will exercise the unit as realistically as possible in all skills and behaviors that need training (MILES is an example of improved effectiveness through improved realism and improved efficiency through the use of less ammunition and less risk of soldier injury).

(3) Evaluation Planning: To provide measurement for exercise evaluation, it is necessary to identify ahead of time the observable behaviors that will indicate the acceptable level of skill attainment. The means to collect the data must be provided, too.

(4) Execution: Carry out the training exercise with sufficient simulation of surrounding units and OPFOR, and sufficient number of observers at key events.

(5) Evaluation: Process and interpret the results. Produce a training summary which includes:

(a) What skills have been demonstrated by the various units.

(b) Comparison of the deficiencies with the required skills outlined in (1) above.

(c) A list of skills that need further training.

This five step cycle is repeated as long as time and resources are available and as long as there are important skills that need improvement.

2.3 Points of Improvement in ARTEP

This report is part of a larger project aimed at improvement and further development of tools for tactical training evaluation. For this reason, we will discuss here in more detail the current methodology and tools applied to the evaluation phase of the training cycle and suggest points of improvement.

Before going into the content of the T&EO, let us suggest a list of objectives that can guide us in evaluating an evaluation system:

(1) Reduce the training and experience required of the training officer and staff needed to attain an acceptable level of training performance.

(2) Aid the officer to collect all the relevant and necessary data without depleting resources, manpower, etc.

(3) Reduce the time needed for and improve the quality of the evaluation process.

These general objectives can be met by providing assistance in the following more specific steps of the evaluation process:

(1) Aid the officer in identifying the relevant data and plan a collecting scheme to obtain the data (observers at the right point at the right time).

(2) Filter irrelevant data from the large amount collected.

(3) Concentrate the relevant data in usable form.

(4) Identify which tasks were accomplished and which were not.

(5) Identify missing tactical skills.

(6) Identify specific behaviors that caused eventual success or failure.

(7) Filter events that are peculiar to the particular exercise run and do not carry over.

(8) Aid in focusing attention on major problem areas demonstrated in the exercise.

(9) Provide tutorial aids for an officer who is not familiar with a given mission, maneuver, or the particular tactical circumstance.

(10) Help to summarize and aggregate the problem areas identified so that the most critical ones can be addressed first.

(11) Help translate the deficiencies discovered into requirements for the next exercise to make it most effective.

The ARTEP system and documents of the T&EO are a major step in the direction of providing a systematic way for tactical evaluation. They provide a training officer with a single set of documents in which he can find what he needs for planning an exercise and laying of an evaluation plan, and also, with a set of performance standards by which he can judge the performance. The T&EO's break down the overall military mission into units of different size and kind and, further, into various tasks that a given unit can be expected to perform. They provide the information for planning and evaluation by associating the following three kinds of tactical information:

(1) Task - what are the component tasks of the given mission, who has to do it, and in what sequence.

(2) Conditions - what are the conditions that must be met for the task to be meaningfully started or conducted.

(3) Evaluation standards - general statements of expected performance levels.

These three components of the T&EO meet in our judgment the essential requirements of a training evaluation aid. They tell the evaluator what had to be done, what were the preconditions that made an action valid and what were the expected performance standards of that task. The drawbacks stem mainly from the delivery media--a large, cumbersome, passive, fixed set of documents.

2.4 Evaluation System Requirements

An evaluation system, based on the ARTEP system, but which overcomes the limitations outlined in Section 2.3, should meet most of the following requirements.

At the outset, the system should meet the organizational requirements: it should reduce the training and experience required of the evaluation officer, partly by giving him aiding of different kinds, and partly by being tutorial, as we will see below. It should aid in collecting and filtering out of data, and it should improve the timeliness and quality of the evaluation process.

In terms of functionality, the system has to be portable so that it can be used in the field, probably even during the exercise. It has to be modular so that different capabilities can be provided for users at different levels, i.e., a company evaluation system should be different in size (but not in concept) from a division's system. The system has to be incrementally modifiable, so that as new tactics evolve and new evaluation methods are developed they can be incorporated into the working systems without too much effort and redesign. Finally, the system has to be integrated, i.e., provide a global solution to all the phases of the training process. This will allow the user to see what evaluation information will be necessary at the planning phase of the exercise and thus assign the resources. The T&EO's provide parts of these integration features.

Now we will present point by point the characteristics of an evaluation system that address the disadvantages of the T&EO's in terms of their medium. We wish to maintain the core concept of the T&EO: that of presenting the military mission as tasks, or conditions, when they have

to be modified or hedged, and explicit performance standards. The presentation of this rich tactical knowledge base, however, should be made more flexible in the following ways:

Selective: Under a given set of conditions, e.g., night attack in a hilly area, only the relevant tasks, conditions and performance standards should be presented to the user. Why clutter him with irrelevant data he has to read and discard?

Active: The system can actively participate in the evaluation process by prompting the user for expected events or conditions, thus helping him in effect by filtering out irrelevant data that may have been collected.

Hierarchical: Because much of the military information is hierarchical (e.g., command hierarchy, task assignment to units, priority hierarchies in resource allocation and in critical tactical skills, and many more), it is logical to tailor the access mechanism in a hiararchical form. The user will be able to get quickly to the right data without searching through long lists of data that are irrelevant at the moment.

Rich cross referencing: In addition to a hierarchical organization along several dimensions, the evaluation system should have cross references to other parts of the tactical knowledge base, together with quick access to relevant data upon demand. This richness in cross connection has to be visible only upon request; but its on-hand availability has a strong tutorial impact. The data is there if needed or requested, and does not stand in the way for an experienced user. He in turn might be interested in an explanation or definition which he is unfamiliar with.

Dynamic: A dynamic media can change its presentation with the changing external situation. In the tactical evaluation task the external situation can change at different levels and a properly designed dynamic system can respond at all levels. In essence, a dynamic media can tailor the T&EO around the particular environment (e.g., day/night, terrain), the particular tactical situation (e.g., a river crossing after the third bounding jump), and the particular sequence of events (e.g., saggar attack after the leading tank got stuck in the mud). It is this lack of dynamic capability that keeps the T&EO replete with superfluous information and the performance standards so general and vague.

Adaptive: An adaptive evaluation system can tailor its responses to a particular user. It can adapt to levels of experience, knowledge, or even style of interaction.

Responsive: At any particular evaluation session, the user may be interested in different aspects of the training unit's behavior, and a responsive system would limit its presentation and questions to those required by the user.

Aggregation and diagnosis: A dynamic active evaluation media can take parts of the evaluation burden off the shoulders of the training officer. It can start by doing bookkeeping chores, then provide simple aggregation of factual data and as the technology develops and improves, it can provide assistance in higher cognitive tasks. The technology of Production Rule models provides mechanisms for assistance to relatively high-level cognitive tasks, and this capability can be built incrementally.

3. THE MODEL

3.1 Overview

During the present effort, a tactical evaluation aid was developed which satisfies the system requirements described in the previous chapter. The development of the evaluation aid was realized by adapting an innovative event-driven model, called a production-rule model, to the requirements of the military environment. The production rule model, its relevance to the military environment and its application to this project are the subjects of this chapter.

3.2 Model Concept

Models are used in all areas of scientific endeavor, and more recently in the business and military domains. The wide usage of the term makes it imperative to define the sense in which we are using it, and what we expect the model to provide.

A model is an abstraction of the subject under investigation. It eliminates as much of the details as possible and focuses attention on the essential concepts, relations, and mechanisms of the subject matter. The model, thus, keeps what is relevant and discards what is irrelevant to allow comprehension of the key behaviors or manifestations that are investigated.

The nature of a model developed for an area of investigation depends heavily on what use will be made of the model. And two models of the same problem, if intended for different uses, can be quite dissimilar. Consider, for example, that a material such as wood when intended to be used in a paper factory is modeled and analyzed by the models and termi-

nology of chemistry. When it is used as a building material it is modeled by physical strength and stress models; and when used for decoration, its artistic color, texture, and warmth features are considered. The same is true about modeling in a tactical situation. Here, one objective is to use the model for training evaluation.

Models can be classified further into prescriptive or descriptive models. A prescriptive model indicates what one ought to do in a given situation. A descriptive model is intended to describe empirically what one actually does. Typically, prescriptive models are the outcome of analytical approaches; whereas empirical approaches generally lead to descriptive models. In theory at least, a prescriptive model may be used either as a guide for a tactical practitioner (a commander) or as a standard against which to assess the extent tactical performance approaches this theoretical optimality. Descriptive models differ from prescriptive models insofar as the modeled units, in the real world, perform in a less-than-optimal fashion. Comparison between prescriptive and descriptive models can be instructive in suggesting the reasons why a unit's behavior is not optimal, and just where the deficiencies occur.

This comparison is the essential point in our approach to modeling and evaluating tactical behavior. We have adapted a model that was developed over the last 15-20 years, under the generic name of Artificial Intelligence, into a prescriptive model of tactical behavior. We then developed an interactive program that could elicit a detailed description of what actually happened in the field during the exercise under investigation. The program compares the actual scenario with the internal prescriptive model, and comes up with the possible discrepancies. These uncovered discrepancies, found by the computer rather than by the user, provide substantial assistance in the task of training evaluation. In subsequent sections we will describe the tactical model

and how it is used in the computer program to produce the evaluation of the tactical exercise.

The dynamic, event-driven nature of tactical engagement (compared to tactical events, such as "enemy opens fire unexpectedly," which determine a separate subsequent course of activities) implies that the production rules model is the most appropriate to represent the tactical knowledge base. This similarity is so close that the computer internal model can use verbal descriptions of tasks and events, a feature that is very useful in explaning to the user what action was expected in response to a given event, or why the computer made a certain inference. Such an explanation is both tutorial and engenders acceptability of the computer based tool.

The production rule model allows a hierarchical, modular and incrementally expandable knowledge base. Thus, the tactical knowledge contained in the system can be developed and expanded in an evolutionary way, with new tactics, or new twists to old ones easily incorporated.

The production rule model can accommodate concurrency where several activities can go on at the same time and impact each other, which is also an important feature of any tactical engagement. Furthermore, the representation of the tactical tasks can be used in a computer based simulation, thus providing the capability to try "what if" scenarios, showing the probable outcome of an alternative tactical choice. Finally, the production rules model can be developed to demonstrate inference capabilities, thus expanding dramatically the flexibility of the man-machine interaction as compared to any other programming techniques extant.

3.3 Production Rule Model

Production rule systems represent a successful approach for knowledge representation and deductive mechanisms. In these systems, the problem specific knowledge is packaged as small modular "chunks" called productions. A production is a rule which consists of a situation-recognition part and an action part. Thus a production is a "situation--action" pair in which the left side is a list of things to watch for in the description of the current state of the world, and the right side is the list of things to do in reaction to these descriptions.

In the case of military environments, an example of productions that guide the commander's actions may be:

IF

AND

Self has aggressive objective
Enemy in defensive position
Self has 3:1 weapon ratio advantage
Self can get artillery support
Enemy defense concentrated in front
There is a flank hidden route

THEN

Perform an attack through the weak flank

The effect of such a production is to respond to the situation when all the aspects combined by the AND are present and change the current action from whatever it was before to "Attack through the weak flank."

In addition to the large set of such productions, the production rule system contains a triggering mechanism that uniformly checks all the productions that apply in a given situation (by testing for the truth of

the left hand side of each production) and applies those that are applicable--causing the situation to change.

The main advantages of the production rule approach are the ease and modularity of the knowledge representation. Consequently, it is easy to elicit information from experts without requiring them to be programmers. In fact, many training manuals are written already in "production rule style." Furthermore, the information is incremental; thus it is easily modified, updated and expanded into new areas of expertise. Also, it usually is argued by production rule proponents that this form of knowledge representation is highly compatible with human cognition, making it a very useful and powerful training tool. For example, suppose a training evaluation model is built as a production rule system. It becomes very easy to communicate with the system and ask "Why have you considered that action improper?" meaning what aspects of the situation require the trainee to initiate an action other than the one he chose to perform.

It can be made specifically apparent where the trainee went wrong. At the same time, this is also a powerful debugging tool allowing experts to tune the system by following its reasoning process and identifying the specific cause for a mistaken conclusion which led to an unreasonable response.

3.4 The Productions

As AND/OR graphs (a graph with nodes combined by logical AND or OR functions), production systems are composed of two parts: the set of productions and a mechanism to find a solution in a given situation. We will discus first a graphic representation of the productions themselves. A simple production specifies a single conclusion which follows

from the simultaneous satisfaction of the situation recognition conditions. Any particular conclusion may spring from any production. The conclusion specified in a production follows from the AND or "conjunction" of the facts specified in the premise recognition part. A conclusion reached by more than one production is said to be the OR or "disjunction" of those productions. Depicting these relationships graphically produces an AND/OR graph. Figure 3-1 shows an AND/OR graph which reaches from base tactical facts (F_i) on the left, through the different productions (p_j), to a conclusion or an act to be taken, on the right side of the figure. Any collection of productions implies such a graph. In Figure 3-1 we used the set of tactical productions given in Figure 3-2. These productions should be taken as an example of the capabilities of this approach.

The arrangement of nodes in this graph focuses on how the conclusion can be reached by various combinations of basic facts. As with ordinary AND/OR trees, a conclusion is verified if it is possible to connect it with basic facts through a set of satisfied AND/OR nodes. Different sets of facts can be used to reach a given conclusion by selecting different branches at OR nodes.

3.5 The Control Mechanism

The control mechanism which utilizes the set of productions takes a collection of known facts about the situation and makes new conclusions according to productions that are satisfied by the initial facts. In operation, the user would first gather up all facts available and present them to the system. The control mechanism will then scan the production list for a production which has a matching situation part, i.e., all the premises in the left hand side are satisfied. This production will be activated and its action side will change the facts

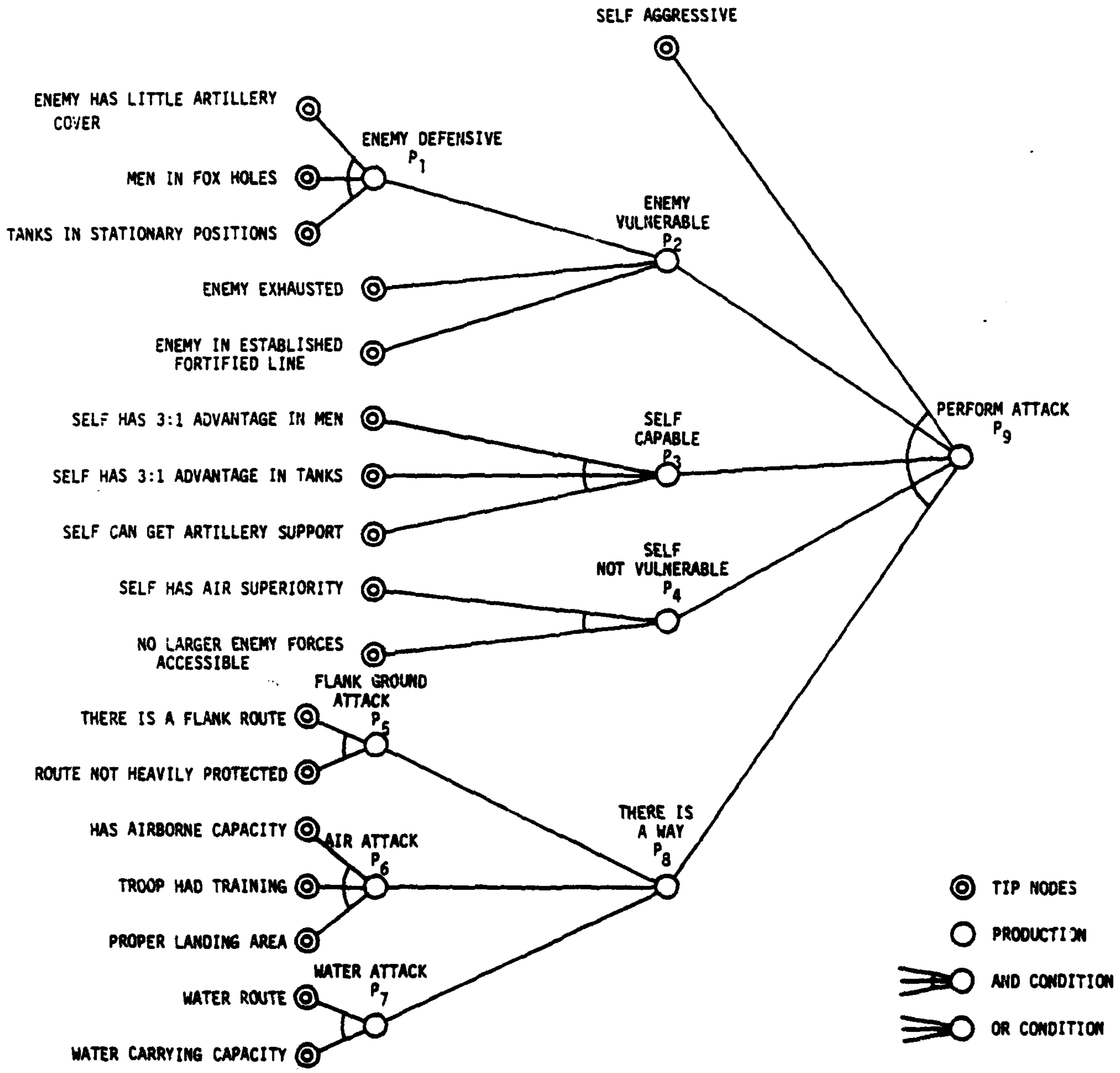

FIGURE 3-1.
AND/OR GRAPH

```
P1

IF
    AND
        Enemy has little artillery cover
        Men in fox holes
        Tanks in stationary positions
THEN
        Enemy defensive

P2

IF
    OR
        Enemy defensive
        Enemy exhausted
        Enemy in established fortified line
THEN
        Enemy vulnerable

P3

IF
    AND
        Self has 3:1 advantage in men
        Self has 3:1 advantage in tanks
        Self can get artillery support
THEN
        Self capable

P4

IF
    AND
        Self has air superiority
        No larger enemy forces accessible
THEN
        Self not vulnerable

P5

IF
    AND
        There is a flank route
        Route not heavily protected
THEN
        Flank ground attack

P6

IF
    AND
        Has airborne capacity
        Troop had training
        Proper landing area
THEN
        Air attack

P7

IF
    AND
        Water route
        Water carrying capacity
THEN
        Water attack

P8

IF
    OR
        Flank ground attack
        Air attack
        Water attack
THEN
        There is a way

P9

IF
    AND
        Self aggressive
        Enemy vulnerable
        Self capable
        Self not vulnerable
        There is a way
THEN
        Perform attack
```

FIGURE 3-2.
PRODUCTION RULE EXAMPLE

known about the situation. In the example given, if P1 were activated, it adds the conclusion that "enemy is in defensive position" to the situation description.

Reasoning from base facts to a conclusion rarely entails using only a single step, however. More often, intermediate facts are generated and used, making the reasoning process more complicated and powerful. One consequence is that the individual productions involved can be small, easily understood, and easily created. Also, note that the intermediate facts added by the lower level productions are tactical facts meaningful to the military users of the system, resulting in many benefits. Using this approach, a training evaluation aid can produce a chain of conclusions leading to intelligent evaluation of tactical actions, even as a trainee makes his actions dynamically, based on changing situations.

In the event that many productions have premises or situation specifications that are satisfied simultaneously, there must be some way of selecting among them. The selection method must be tailored to the specific application area. Some of the popular selection methods are:

(1) All productions are arranged in one long list. The first matching production is the one used. The others are ignored.

(2) The matching production with the toughest requirements is the one used, where "toughest" means the longest list of constraining premises or situation elements.

(3) The matching production most recently used is used again.

(4) Some aspects of the total situation are considered more important. Productions matching high priority situation elements are privileged.

So far, the deduction oriented production system is assumed to work from known facts to new, deduced facts. Running this way, a system is said to exhibit "forward chaining." But "backward chaining" is also possible, for the production system user can hypothesize a conclusion or a desired final state and use the productions to work backward toward an enumeration of the facts that would support the hypothesis. For example, (see Figure 3-1) in the case of an army commander, the system can start from the mission, e.g., attack enemy. Then chaining backward from (P9), it will conclude that it has to achieve self-capability. This can be achieved by providing personnel, tank and artillery advantage over enemy (P8). Thus, by a small change of orientation, the same set of productions was used backwards. Knowing that a deduction-oriented production system can run forward or backward, the question of which is better is decided by the purpose of the reasoning and by the shape of the problem space. Certainly, if the goal is to discover all that can be deduced from a given set of facts, then the production system must run forward. The production system can run forward from all premise elements as long as suitable productions exist. Using sensory systems to supply more facts is necessary only when no productions apply, and no conclusion has been reached. On the other hand, if the purpose is to verify or deny a particular conclusion, or reach a desired situation through a sequence of actions, then the production system is probably best run backward from that conclusion. Avoiding needless fact accumulation is one result obtained; indeed, no irrelevant facts need be checked at all.

Another method for deciding a preference for either forward or backward chaining is illustrated in Figure 3-3 by the use of two symmetric situa-

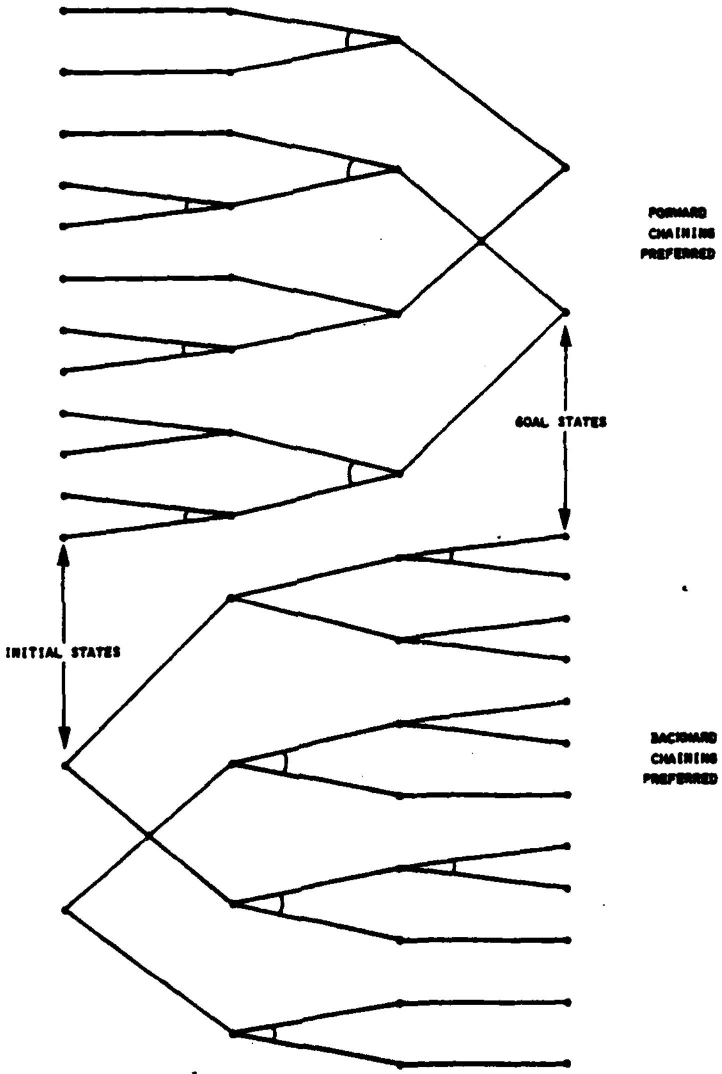

FIGURE 3-3.
AND/OR GRAPH SHAPES FOR FORWARD OR BACKWARD CHAINING

tions. All possible states are represented along with the operations that can change one state into a neighbor. In the first situation shown, forward chaining is better because there is a general fan-in from the typical initial states toward the typical goal states. In this way, it is hard to get into a dead end. In the second situation, the shape favors backward chaining since there is fan out.

3.6 Advantages

The following advantages are associated with production rule systems:

(1) Production systems provide a powerful model of the basic human problem solving mechanisms. This results in easy expert elicitation, user communication at the comfortable level of military tactical concepts and terms, easy trouble-shooting, and good training capability.

(2) System states are meaningful to users, debuggers, etc.; thus an evaluation can be made on the tactical level rather than in the computer implementation level.

(3) Production systems enforce a homogeneous representation of knowledge, effectively separating the static data representation from the uniformly applied evaluation mechanism.

(4) The control mechanism is simple and explicit on what to do next. It is clear from the current state what productions are available.

(5) Production systems allow incremental growth through the addition of individual productions and without changes necessary to any others.

(6) Production systems allow unplanned, but useful, interactions which are not possible with control structures in which all procedural interactions are determined beforehand. A piece of knowledge, or a combination of such, can be applied whenever appropriate, not just whenever a programmer predicts it can be appropriate. This can lead to highly intelligent performance by systems with a surprisingly small (several hundreds) set of productions.

(7) Providing explanation capability to the system is natural to implement. When some decision is made, the system can present the sequence of productions that led to that decision, thus affording its "reasoning" about the situation.

(8) The production rule approach is as general as any other method based on the state space model.

(9) Productions can be quantified with probability information leading to applicability in decision making and risk evaluation.

3.7 Disadvantages

Some of the advantages of the production rule approach can become disadvantages if care is not exercised in the design process:

(1) Maintaining focus of attention: It would seem that production rule systems allow knowledge to be tossed into the system homogeneously and incrementally without worry about relating new knowledge quanta to old. Thus, by relinquishing control, such systems allow unimportant productions to usurp center stage from more important productions, leading the process astray.

(2) Size problems: One particular problem is that production systems may break down if the amount of knowledge is too large, or when the number of productions grows behyond reasonable bounds. The advantage of not needing to worry about the interactions among the productions can become the disadvantage of not being able to influence the interactions among the larger number of productions.

The possible solution, of course, is to partition the facts and the productions into subsystems such that at any time only a manageable number are under consideration. Within each subsystem, some productions may be devoted to arranging transfer of information or attention to another subsystem. Curiously, some users of Hewitt's ACTORS language produce programs that have a strong resemblance to systems of communicating production subsystems.

The solution, however, goes against one of the main advantages of production rule systems, namely, modularity and independent control. If control guiding productions are added, we again have the problem of explicitly directing where control should go.

(3) Global effects: It is awkward to represent global effects using production rule approach. Here, again, the modularity of the productions requires that if some global effects take part in many productions, it is necessary to duplicate the whole set of productions which behave differently for each different state (e.g., different weather).

3.8 Model Essentials

The production rules approach is advantageous wherever the application can be naturally represented in the "Condition - Action" format. Moreover, the direct representation of tactical tasks and events provided by the model and its event driven character lend themselves naturally to a tactical evaluation tool. Given the events that actually happened in the ongoing exercise, the computer can access its production rules-based tactical knowledge base and compare the actions taken by the training unit with the proper response to these events. In the future NTC environment, the data collection will be at least partially automated, but in this small demonstration program, the evaluator has to provide the data himself, both the factual data about the task that the unit performed and the tactical event that occurred, and evaluative information on how well the unit performed a particular task. It must be noted, however, that even if most of the raw data will be collected automatically, its interpretation and evaluation will have to be done by experts. The same physical move of a tank down a hill may be considered correct or incorrect depending on subtle differences in time, location of the OPFOR or relation to other forces. This evaluation will still be helped by an evaluation aid such as the one demonstrated here.

The objective of real-time evaluation of a unit's tactical performance during training is to ascertain:

(1) What tasks the unit performed correctly?

(2) Where did it respond incorrectly to events and enemy actions?

(3) What action did it take at the wrong time?

(4) On what performance measures did it not meet expected performance standards?

(5) What class of skills did it manifestly not possess?

A prescriptive model that is useful to obtain answers to these and similar questions must contain information about all these aspects of the tactical battlefield and missions. It is further advantageous if the model uses the same terminology used by the military personnel so that they can interact with it on their own terms.

During this initial project we have adapted the production rule model to serve as a language to express land tactical missions and a process for its manipulation. Using this model the tactical mission is described as a hierarchy of tasks and subtasks performed by a unit and its components. The tasks correspond directly to the military missions. One task hierarchy will describe a "Hasty Attack," another will present a "Defense" and so on for all missions of a unit. The task is described in detail in terms of the subtasks that make it up. The key elements in a tactical mission are: what has to be done, and what to do when some event occurs. Thus, our model is essentially made up of tasks and transitions among them which are caused by specific events.

Figure 3-4 shows a diagram of a hypothetical mission. At the top level (if we are not interested in more detail) the mission is described as a block with a statement of what it is designed to accomplish. The graph below the mission block indicates the tasks that have to be accomplished in order to accomplish the mission, in what sequence they have to be performed, and what to do when some specific events occur.

Each block in the figure indicates a tactical task (e.g., move to the next overwatch position) and each arrow in the figure stands for a production rule. Stated in words, the arrows with the bars of events E1 and E2 on them state:

If you are in the starting task of the mission X
then
When event E1 occures start Task 2.
When event E2 ocures start Task 2'.

The unit starts the mission by completing the "starting task" (indicated by a dashed pointer emanating from the mission block). This task may terminate naturally when completed (Event_1) or when some extraneous event (E_2) occurs. The proper thing to do upon natural completion is to perform Task 1. When event E_2 has occurred, however, the proper task is Task 2. It is important to note here that the diagram which describes the mission in Figure 3-4 (We call it the schema of the mission.) does not describe one scenario of the mission. It represents many possible scenarios, each differing in the details of the timing and events that occurred in the particular scenario. Each specific scenario is just one possible path through the schema from the starting task to one of the several final tasks. Figure 3-5 shows one correct scenario through the schema of the mission in diagram 3-4. The scenario is correct in the sense that the unit terminated each task upon the occurrence of the specified event and that each event caused a transition to the expected next task.

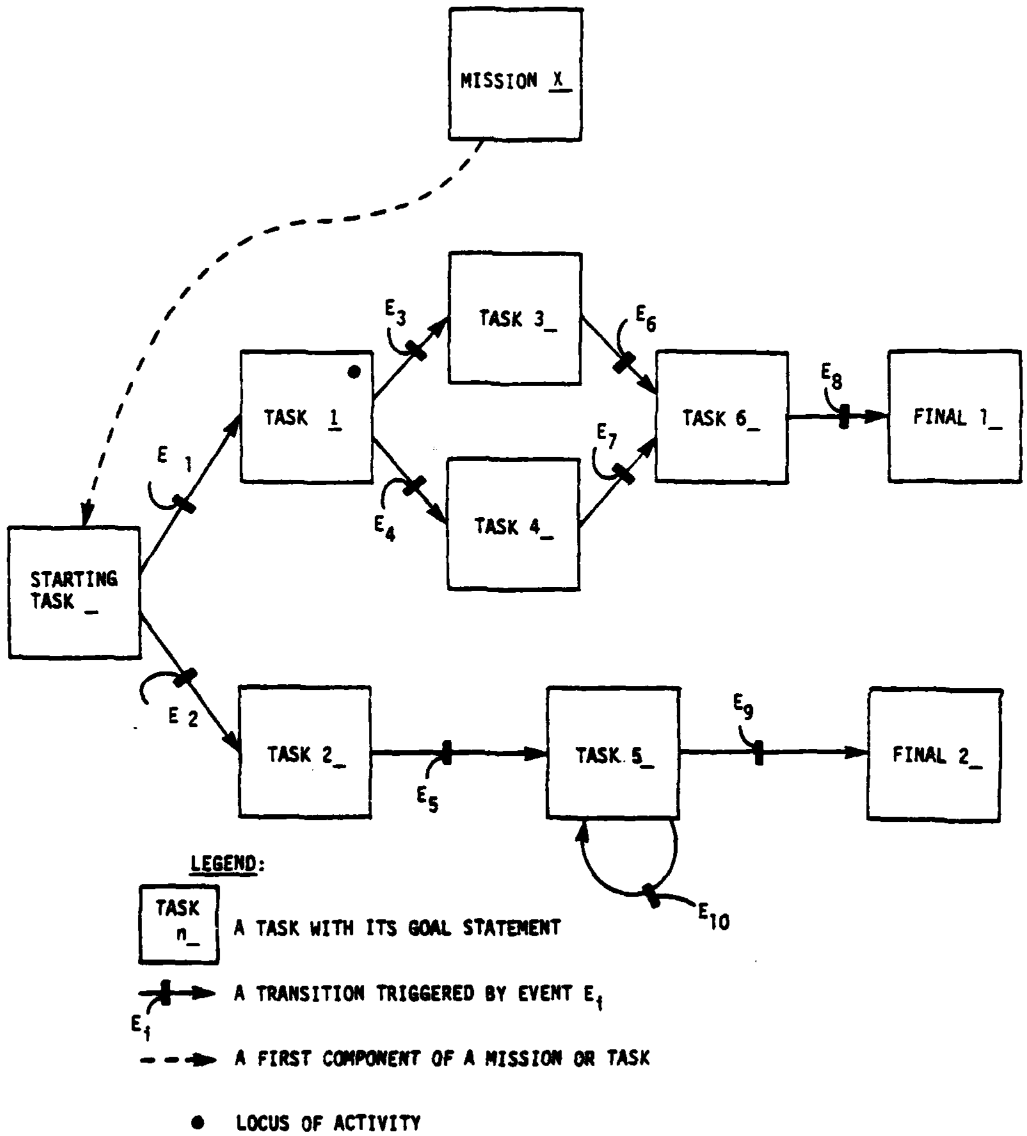

FIGURE 3-4
A TYPICAL MISSION

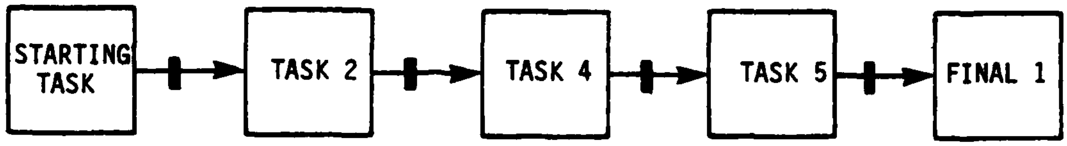

FIGURE 3-5
A SPECIFIC SCENARIO OF THE MISSION IN FIGURE 3-1.

From this simple example we can see the main mechanism of the evaluation process that can be constructed on this model. By taking a detailed schema description of the mission under training, eliciting from the evaluating officer the particular scenario as it unfolded during the exercise, and comparing one to the other, the system can identify the following:

(1) Which tasks were attempted.

(2) Which were accomplished.

(3) Which events were detected by the unit.

(4) Which significant events were not recognized by the unit.

(5) To which events the unit responded properly.

(6) To which it did not respond properly.

Answering these questions, and generalizing from the answers, can provide an approach to identifing deficient skills in terms of deficient behaviors in the field.

To summarize our approach, a prescriptive event-driven model (developed from military experts, T&EO, and other manuals) was constructed to represent a simple mission in all its possible unfoldings. This model was further translated into a form internal to a computer and directly accessible by the evaluation program. The program compares this prescriptive model witn the actual succession of tasks performed by the unit during the exercise, and generates a very specific evaluation of what the unit performed correctly and what it did not. This process is

ongoing dynamically on an interactive system so that the additional advantages of a computer's fast data retrieval and information processing capabilities can be utilized. The schema can be much more complex (in terms of the number of alternative paths and events considered at every task) than that presented to the user in any specific scenario. The user is presented only with what is relevant to the particular sequence of events that happened in the exercise. The evaluation information shown to the user is thus dynamically tailored to the particular exercise run. By having this selective capability, the description provided for any particular task can be made more specific than that feasible in T&EO. This is because the same T&EO has to be relevant to all possible exercises of a given mission. Furthermore, the events that select a particular path through a schema can depend on other tactical considerations, such as weather, ground composition, or topographical features. The tasks contained in such a schema can be correspondingly more detailed and the evaluation provided more specific.

3.9 Model Content

In addition to the structural components of the model described in the previous section, the model contains additional elements that are useful in the evaluation process. These are descriptions of the tasks, standards of performance, other performance measures, and resource utilization information. These are provided as side benefits to the easy accessibility of the structural information. At any given point, all these pieces of evaluative information are accessible to the user and to the evaluating program with which he interacts. Again, similar information is available in the T&EO, but the model approach makes it more specific and dynamically adaptable to the particular scenario. The extra information is:

Task Description. A definition of the task, with more explanatory information (tutorial) available upon request.

Standards of Performance. A specific statement of the minimal level of performance expected of the unit in the particular task which, if not attained, the unit is considered not to have accomplished the task.

Performance Measures. A list of other performance measures that can help evaluate the performance of the task. They contain also the range of values of the particular parameter that is considered to be an acceptable level of performance. For example, when talking of a "move to contact," the expected average speed is 15-25 mph. This kind of evaluative and tutorial information is available and accessible with each task.

Resource Utilization. A list of resources, tanks' ammunition, and personnel that may be used in a given task is also provided so that resource utilization can be evaluated.

All these pieces of information provide a comprehensive set of accessible, evaluative data against which the actual unit's performance can be compared. The comparison can be done either by the evaluating officer, or, as the evaluative mechanism increases in sophistication, the burden of handling the details can shift to the computer.

4. THE TACTICAL SCHEMA

4.1 Overview

The model described in Chapter 3 can be considered as a language in which tactical missions are described, and which a computer uses dynamically. This point is one of the most important issues to understand. The evaluation aid works by using an internal, detailed description of the particular maneuver that is being evaluated. This description contains an embelished version of T&EO information, with an added feature--a computer program that communicates with the user and can access explicitly each part of the description, and "know" what it represents.

The computer program can find by itself what is the next task expected of the unit and the possible events that may influence that task. When the user inputs the sequence of tasks that actually occurred in the field (referring to them by task names), the program can compare this input to the expected actions, tasks, and events, and use the cummulative information (including the basic descriptions) to aid in the evaluation. To summarize, the internal "knowledge" of the scenario is the basis for the dynamic evaluation process which this demonstration exemplifies.

The concept of schema used in this chapter is closely related to, but still somewhat different from, the "scenario" used by current training systems and personnel. Because of the static nature of the training materials and the limited variety in terrain, the training scenario as it is now used is very rigid and represents only one possible sequence of events. The engagements, events, etc. are all prearranged. The schema described in this chapter is more general. It represents the total unfolding of many possible events.

The event-driven character of the model means that the internal description of a particular task includes several possible events that might terminate the task. In other exercise runs, the triggering event may differ and the system will respond accordingly. The schema can be considered a description of a family of different scenarios, all derived from the same general mission. The details of the system response in any particular exercise run depends on the particular sequence of events that actually occurred. In a word, the system adapts dynamically its general internal description to the particulars of the exercise run being evaluated.

This chapter will present the particular schema, a Hasty Attack by a tank platoon, that is the tactical content of the current demonstration. In essense, 'Move to Contact' in this mission is the only tactical knowledge the system possesses at this stage. It is used here to show how such a computer-based description can be used interactively to evaluate a particular exercise. The chapter describes all the tasks, actions and events that make up this scenario but does not present the details of the evaluative information that is included in the computer program itself.

4.2 The Scenario's Schema

As was discussed in the previous section, this section will describe the content of the knowledge-base which represents the implemented part of the Hasty Attack mission of a tank platoon. Note that the same schema can be applied on different terrains, with the OPFOR engaged at different times and the river crossing task eliminated or included at different points in time. Each different scenario would mean a different run through the Hasty Attack schema, i.e., it would be represented as a different path, possibly with a different number of cycles through the

bounding loop, and through the network of tasks and events.

Figures 4-1, 4-2, and 4-3 show the part of the schema that captures the Bounding Overwatch phase of the "Move to Contact" task within the Hasty Attack mission. This hierarchical relation is represented in the top part of Figure 4-1. A box in the diagrams represents a task with its name on it (We added numbers in Figure 4-1 for ease of reference.) with a box with three kinds of arrows emanating from it. An arrow with a cross bar represents a terminating event for the task with the head of the arrow indicating the task the unit is expected to start with the occurrence of that event. The double lined arrow leads from the task to the first subtask in a more detailed description of its procedure.

Thus, a "Move to Contact" is made up of the following subtasks that are to be accomplished in sequence: (1) Tactical Road March, (2) Travel Off Road, and (3) Bounding Overwatch. The two loops of tasks at the bottom are an elaboration of the Bounding Overwatch maneuver. The dashed arrow is used dynamically during the interaction, and it points to the specific subtask the unit is "currently" performing (the task being evaluated by the officer). In addition, each task contains a list of several performance measures relevant to the particular task. These are shown schematically as horizontal bars under each box. More general performance measures are usually associated with tasks that are placed higher in the hierarchy.

Let us now follow the schema in detail. Starting from the top, we see that the mission at the top level task is a "Hasty Attack." The level below this one represents the major components of this overall mission, among them is the "Move to Contact" [2]. Other components are ignored in this example, but we can see an enter and an exit event. The platoon enters the task [2] when the D time arrives (an event) and leaves it

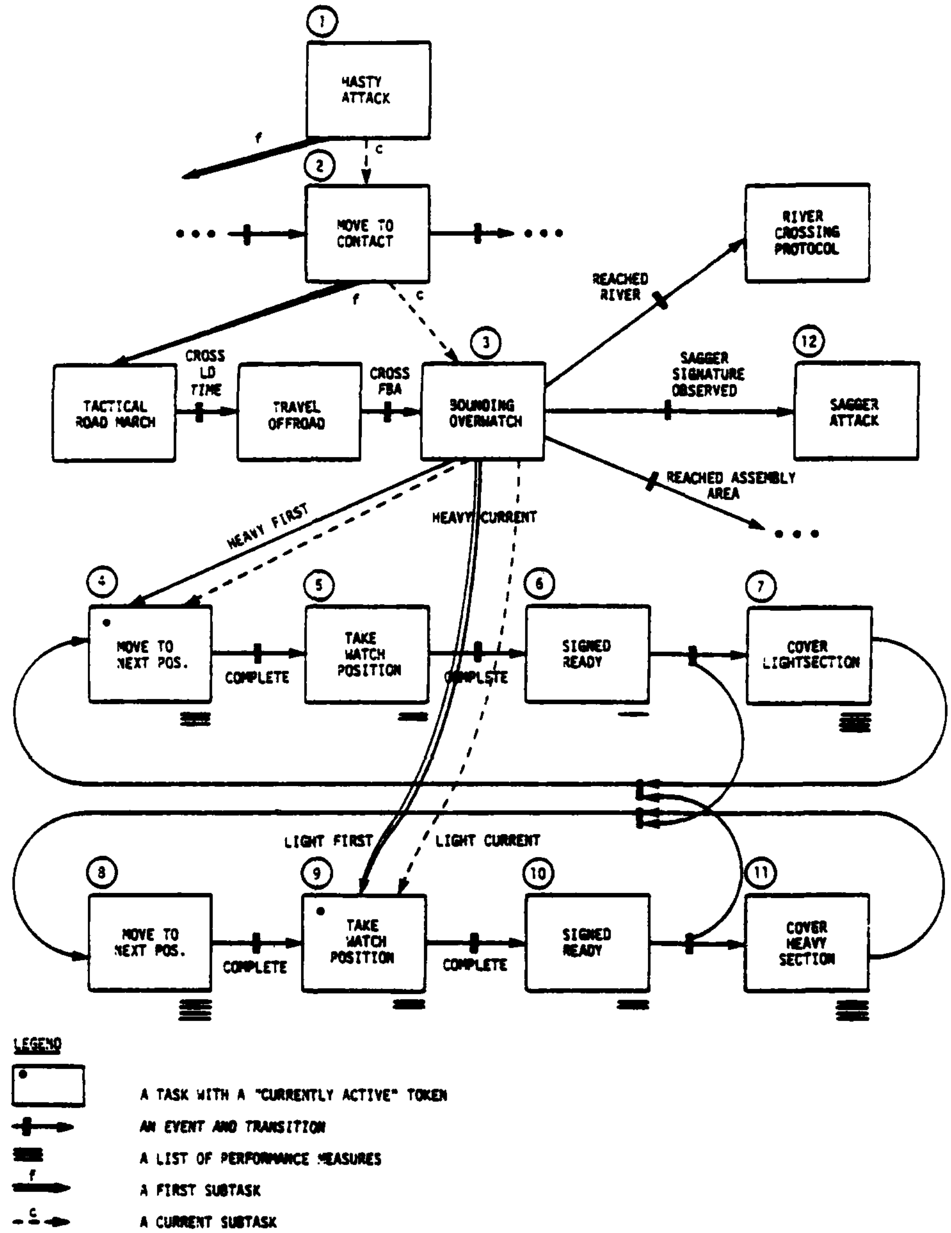

FIGURE 4-1.
THE IMPLEMENTED PART OF THE HASTY ATTACK SCHEMA

SAGGER ATTACK DETAILED ACTIVITIES

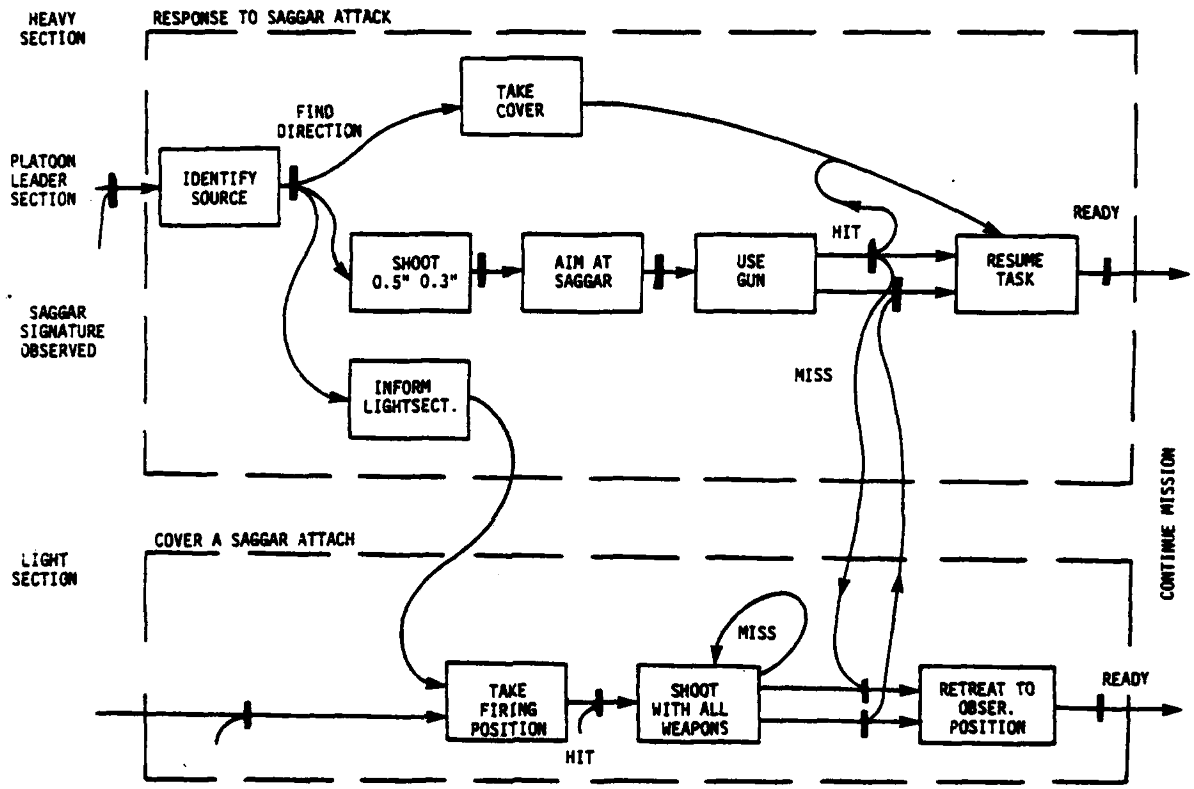

FIGURE 4-2.
SAGGER ATTACK DETAILED ACTIVITIES

RIVER CROSSING DETAILED ACTIVITIES

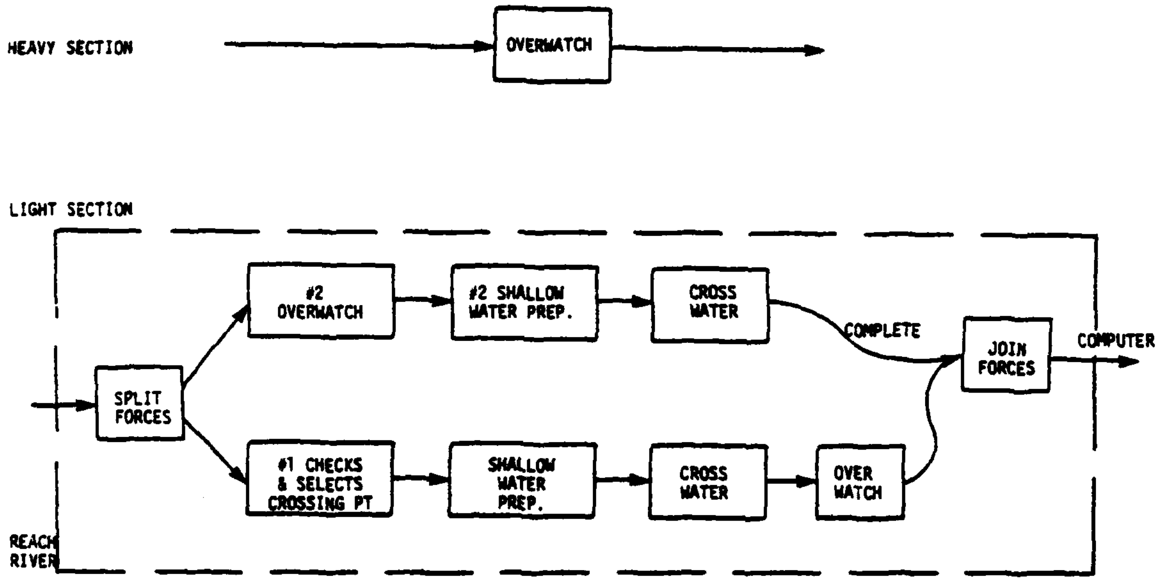

FIGURE 4-3.
RIVER CROSSING DETAILED ACTIVITIES

when it successfully reaches the assembly area and starts to get ready for an assault on the target. We concentrate on the Bounding Overwatch maneuver, and the bottom level shows a detailed breakdown of this maneuver into the sequence of activities done by the heavy section and the light section.

The nature of the Bounding Overwatch maneuver is such that the unit breaks up into two sections where each alternately takes a cover position, gives cover to the other section while that one is in motion, and then moves to the next position on its own. All these elements are represented by the schema in Figure 4-1. We can see that the unit breaks up into two sections by the fact that task [3] has two "first" tasks--one labeled "heavy first" and the other "light first." The sequence of subtasks for each section is identified, including: (1) move to next position [4], (2) take watch position [5], (3) signal ready [6], and (4) cover other section [7]. The difference is that the light section starts with the task "take watch position" while the heavy section starts with "move to next position." This alternating cycle continues as long as the terrain and the distance to the target require.

Here we can see that the schema can adapt dynamically to different terrain and tactical conditions. We can see here another important feature. Notice that the light section's transition from task [11]: cover heavy section, to [8]: move to next position, has two arrows entering the event bar; tactically, this means that the light section has to wait for the ready signal from the heavy section to start its transition to task [8] again. This is the method by which one subunit or the OPFOR can trigger a response by another subunit.

The normal termination of this bounding cycle is when the event "reached assembly area" occurs (the transition emanates from task [3]). Again,

the significance of this is flexibility; the model does not indicate how many cycles there should be in the bounding process; it just shows that the Bounding Overwatch task terminates when the unit reaches the assembly area where it will start the next task.

We see also two other events that may terminate task [3]; they are: "Sagger Attack" and "River Crossing." These are also examples of a dynamic flexibility of the model. In the first place, a "Sagger Attack" can occur at any time during the Bounding Overwatch, but only when it does will the transition occur. In the second place, there may be many potential maneuvers hanging from any task box, but none of them is shown to the user unless the triggering event specific to them occurs. Thus the system can be much more complex and detailed than is shown to the user in any given scenario, and the burden of filtering the irrelevant details is carried by the computer. The user receives only a response tailored to the specific events that occurred or that he contemplates at a given time.

Figures 4-2 and 4-3 show a breakdown of these two tasks, "Sagger Attack," and "River Crossing" into the detailed actions, responses, and outcomes of the two sections making up the platoon.

Figure 4-1 the [River Crossing] and [Sagger Attack] are shown as simple blocks. Figures 4-2 and 4-3 expand them with full details of the component task for the heavy section and the light section. The progress of tasks and events is evident from the figures because all the terminology included in the figure is the military terminology, and the breakdown into subtasks corresponds closely with that used by the military users.

4.3 A Specific Scenario

Figure 4-4 presents a "topo" map of a tactical training area for a tank platoon. A river is indicated, and there are six preselected acceptable observation points. This is the terrain for a specific scenario that is derived from the schema discussed in Section 4.2. Figure 4-5 a and b show the specific movements of the two sections of the platoon as they performed in a particular exercise run. We can see that the platoon performed two complete bounding moves without any unusual event, then the heavy section was engaged by a sagger position on its left, the light section gave cover and hit the sagger position. Then, the heavy section commenced its motion. The light section reached a river and had to perform at that time the River Crossing maneuver. It is clear that this specific scenario and many variations thereof can be derived from the schema in Section 4.2. The dynamic flexibility and adaptability of the event-driven model is evident.

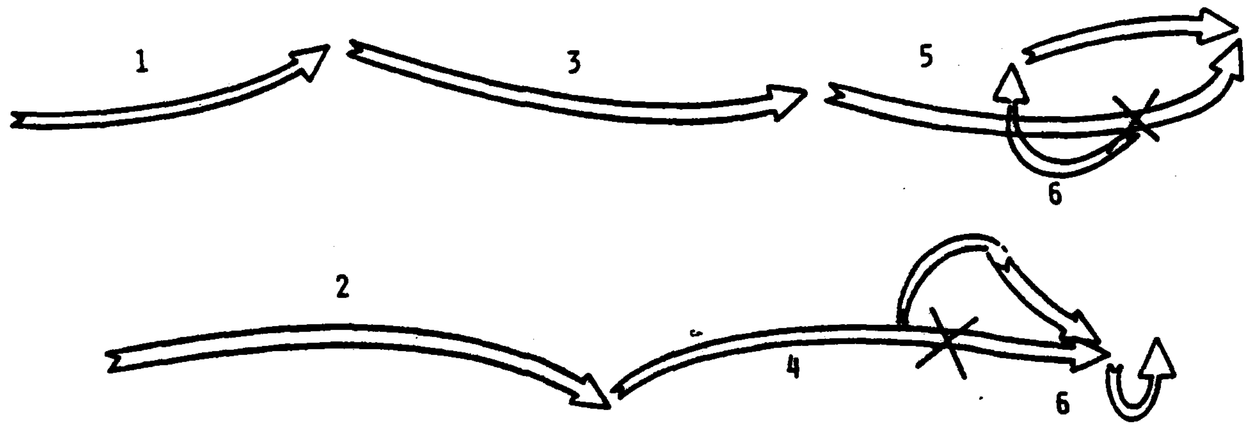

FIGURE 4-5A.
SCHOOL SOLUTION OVERLAY

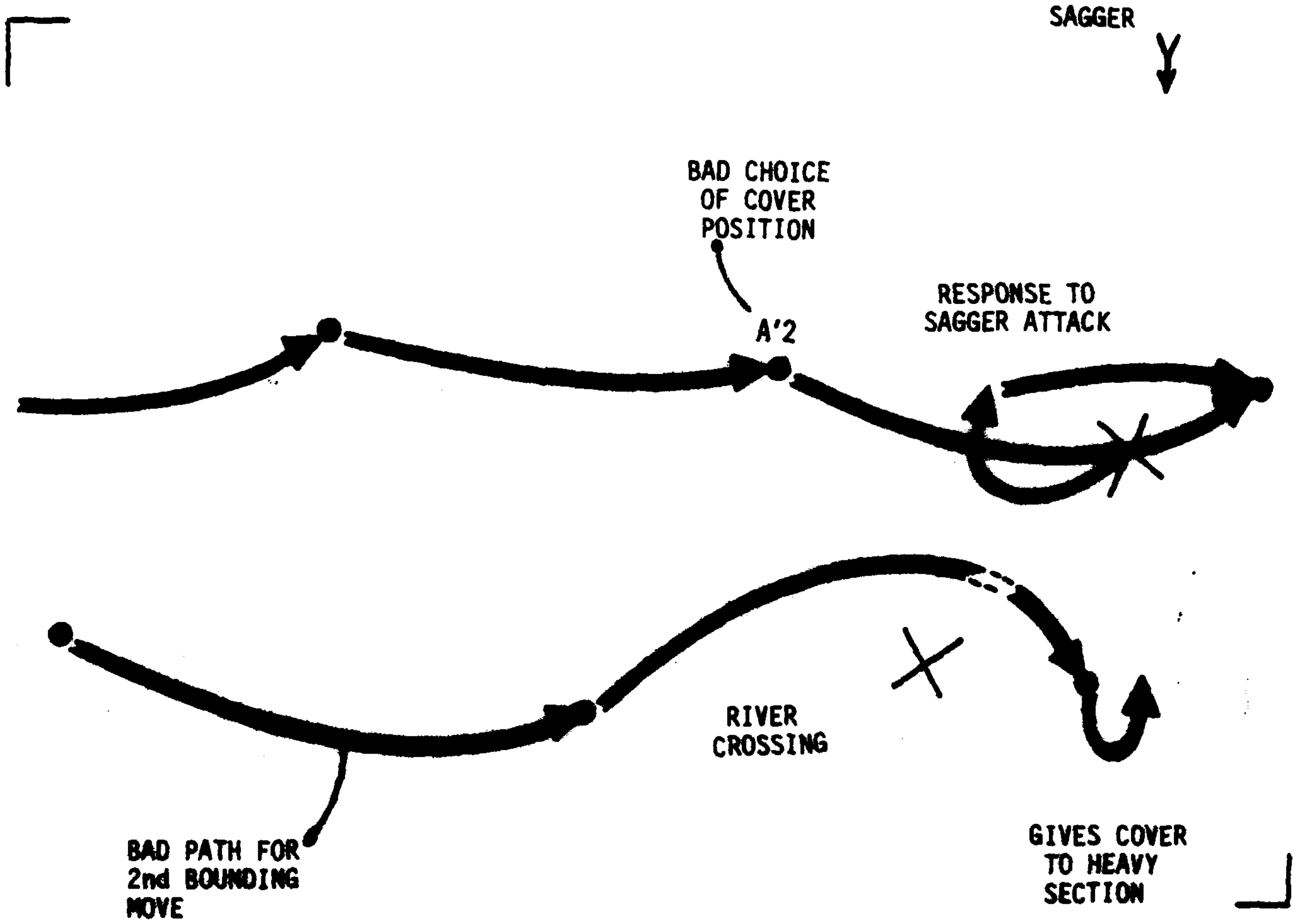

FIGURE 4-5B.
TRAINEE'S ACTIONS OVERLAY

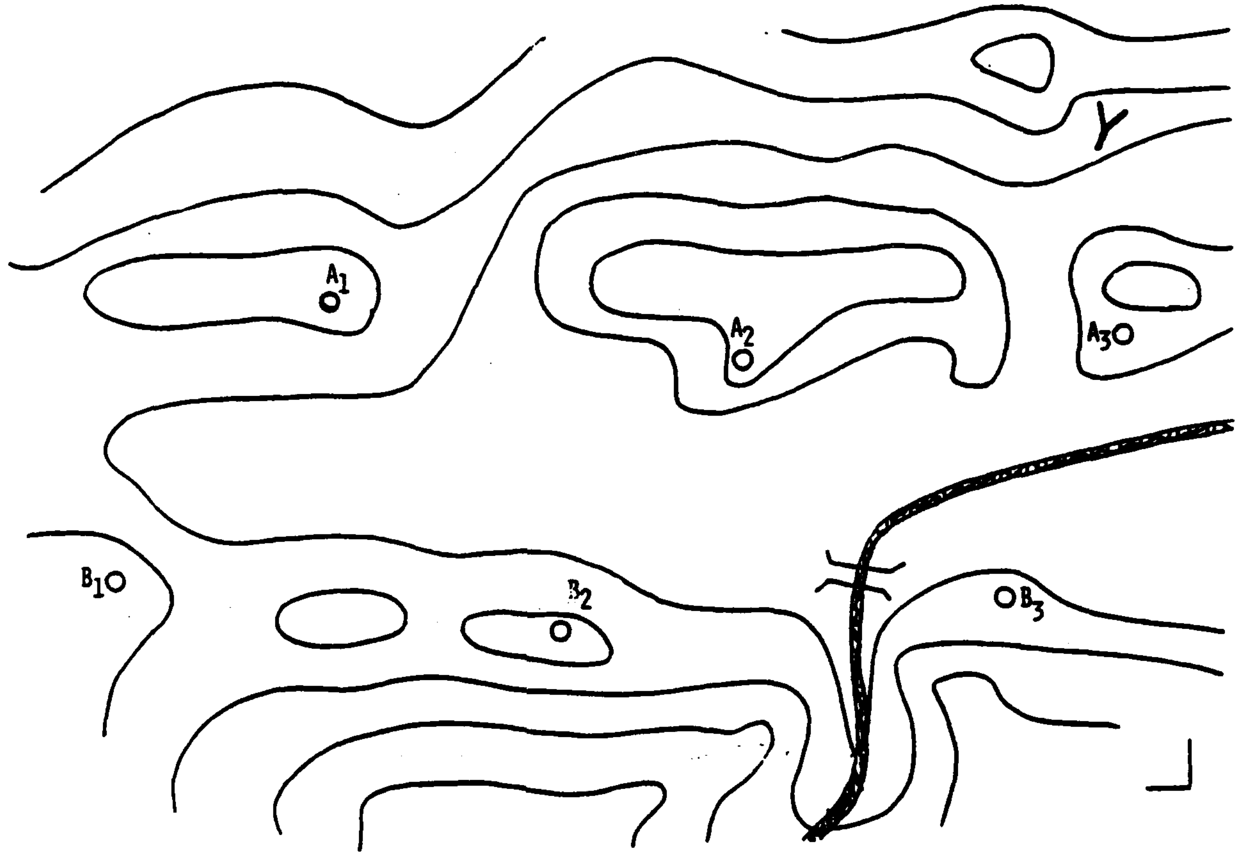

FIGURE 4-4.
TOPOGRAPHIC MAP OF EXERCISE AREA

5. DEMONSTRATION PACKAGE

5.1 Overview

This chapter describes the actual implementation of the demonstration package on a portable computer. It is a concept demonstration of a possible future computer-based system to be used by a training officer in evaluating a tactical exercise.

The package is a 1000-Line PASCAL program. It is written under the UCSD PASCAL operating system, which makes it easy to transfer to most of the available microcomputers. Bear in mind that the power of these microcomputers increases substantially, approximately doubling every year. It can thus be expected that in a period of five years the computing power of current medium and large computers will be available in microcomputer size.

PASCAL is the closest language to ADA, the future standard language of the DOD. It is also available now, together with a flexible operating system (UCSD PASCAL) on the microcomputer on which the demonstration was to be programmed. In future developments, when more powerful CPUs will be available, and especially if complex inference mechanisms will be required, the language, LISP, mignt be considered as a better implementation choice.

The specific hardware used in the demonstration was as follows:

(1) Digital Equipment's 16 bit LSI 11/2, central processing unit.

(2) 64000 Bytes of Random Access Memory.

(3) 4 Serial Input/Output Chanels.

(4) Dual floppy disks with 1.2 Million Bytes of backup memory.

(5) PASCAL Language.

(6) Memory used by the program - 20000 Bytes.

The remainder of this chapter will describe the overall system process and present a sample interaction with the user. Then a sample evaluation summary report will be discussed; the report shows the type of evaluation summarization that can be provided even at this simple level of implementation. Finally, the main programs and corresponding data structures are presented.

5.2 The System Process

The system (hardware and software) developed in this pilot effort is designed to simulate an evaluation tool that would be used by a training officer for post-exercise evaluation. It helps him identify all that happened in the exercise, including the tasks performed by each unit and the expected standards of performance for each task. It helps him evaluate other performance measures associated with the task. Then, after going through the whole exercise in this fashion, it produces a detailed summary of the findings.

Figure 5-1 is a simplified flow diagram of the system process. On the right side of the figure, the main functions of the program are presented as blocks, and the simple arrows show the control flow between them.

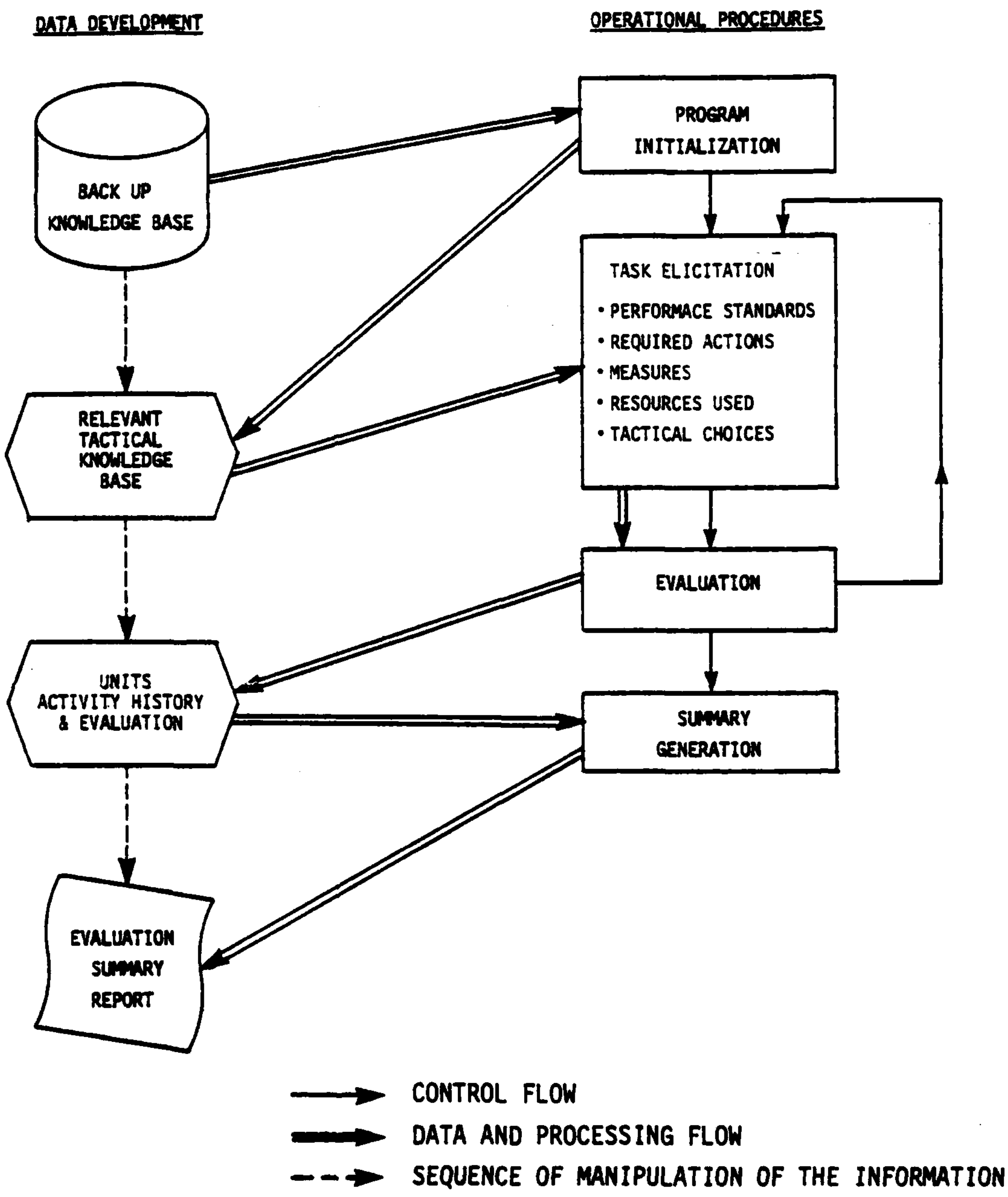

FIGURE 5-1.
OVERALL SYSTEM'S PROCESS

On the left side, from the top down, we see the sequence of steps that the information goes through until a final Evaluation Summary Report of the particular exercise is produced. The double arrows between the two columns show the data flow, that is, how each functional program takes input from a previous state of the data development and produces the next state.

The PROGRAM INITIALIZATION function sets up the relevant knowledge base for a particular exercise. It asks the user for the mission he wants to evaluate, the type of evaluation he will want performed, and what his main concerns are. Using the training officer's responses, the program brings from the large back up disk only the information relevant to the mission at hand.

The TASK ELICITATION block is the main system process. For each successive task performed by each subunit in the exercise, it elicits the following information:

(1) Was the task completed at all?

(2) Were the performance standards (presented by the program) met according to the other user assessment?

(3) Did the unit perform all specific actions associated with the task (These can be considered a form of performance measures.)?

(4) What were the unit's performance levels on all of several performance measures associated with the task?

(5) What were the resources used by the unit during the task performance, including casualties?

(6) What was the tactical event that caused the termination of the task, and when was the unit's next task commenced?

This functional block is organized as a loop and the number of times it goes through this cycle is determined by the number of tasks performed by the unit during the exercise. For each task, the program presents to the user, interactively, the relevant performance measures that it takes from the tactical knowledge base, obtains his inputs, evaluations and judgments, and produces a record of the unit's activity (sequence of tasks), and evaluation of each according to the performance measures presented. The performance measures are structured into a hierarchy of performance measure classes, and all the evaluations that fall in each class are accumulated as the elicitation progresses.

After all the data is assembled, the SUMMARY GENERATOR is called. It goes through the performance measures one by one and produces a summary of where the unit failed to meet them. It also presents tasks that were not completed, key events that occurred which the unit did not respond to properly and non-events that it did respond to appropriately. All this information is useful in focusing the evaluator's attention on the areas where deficiencies in the units' skills have been demonstrated in the exercise.

5.3 Interaction Display

Figure 5-2 illustrates the CRT display as it is seen by the user during the dialog. The dashed lines are imaginary borders that outline where each type of information is displayed. The principles of consistent and

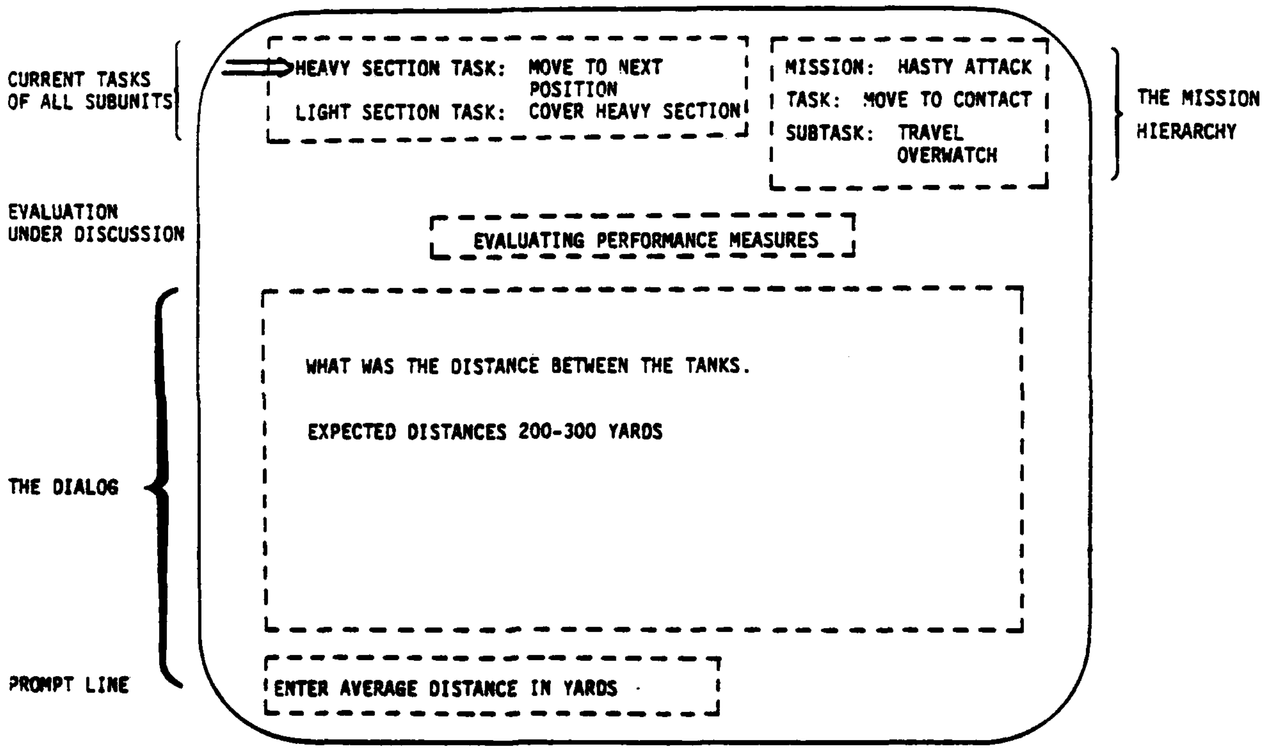

FIGURE 5-2.
A SAMPLE INTERACTION DISPLAY

informative man-machine dialog were strictly maintained. Each area of the screen presents the same kind of information at all times. At the top right corner, the mission hierarchy is given, so that the user always knows the mission and the hierarchy of tasks under it down to one level above the task he is communicating with. This provides a global view. At the top left are the specific tasks "currently" (the time in the exercise) under discussion. The subunit considered at the moment is highlighted with an arrow. At the center of the screen, the type of evaluation currently being done is presented. The rest of the lower part of the screen is used for the free format dialog with a prompting line at the bottom indicating the specific kind of answer expected.

5.4 A Sample Evaluation Summary Report

The final output of the aiding system is an evaluation summary report. This report can be done at different levels of summaries. Figure 5-3 is a scheme to classify the dimensions along which a summary can be generated. According to this scheme, the three major dimensions include: (1) weapon use, (2) tactical movement, (3) communication. Within these classes, we may have significant subclasses, e.g., use of the tank's main gun versus the use of personal weapons and instances of the manifestation of a particular skill.

The summary provided by a system depends on the key issue it tries to address. In this system the main point of demonstration was the system's capability to follow interactively the sequence of tactical tasks in the exercise, to note the key event and to help evaluate the training unit's response to them. In this way, the system can help identify the key point that led to eventual success or failure. This evaluation of intermediate choices is diametrically opposed to strict outcome performance measures that are used currently.

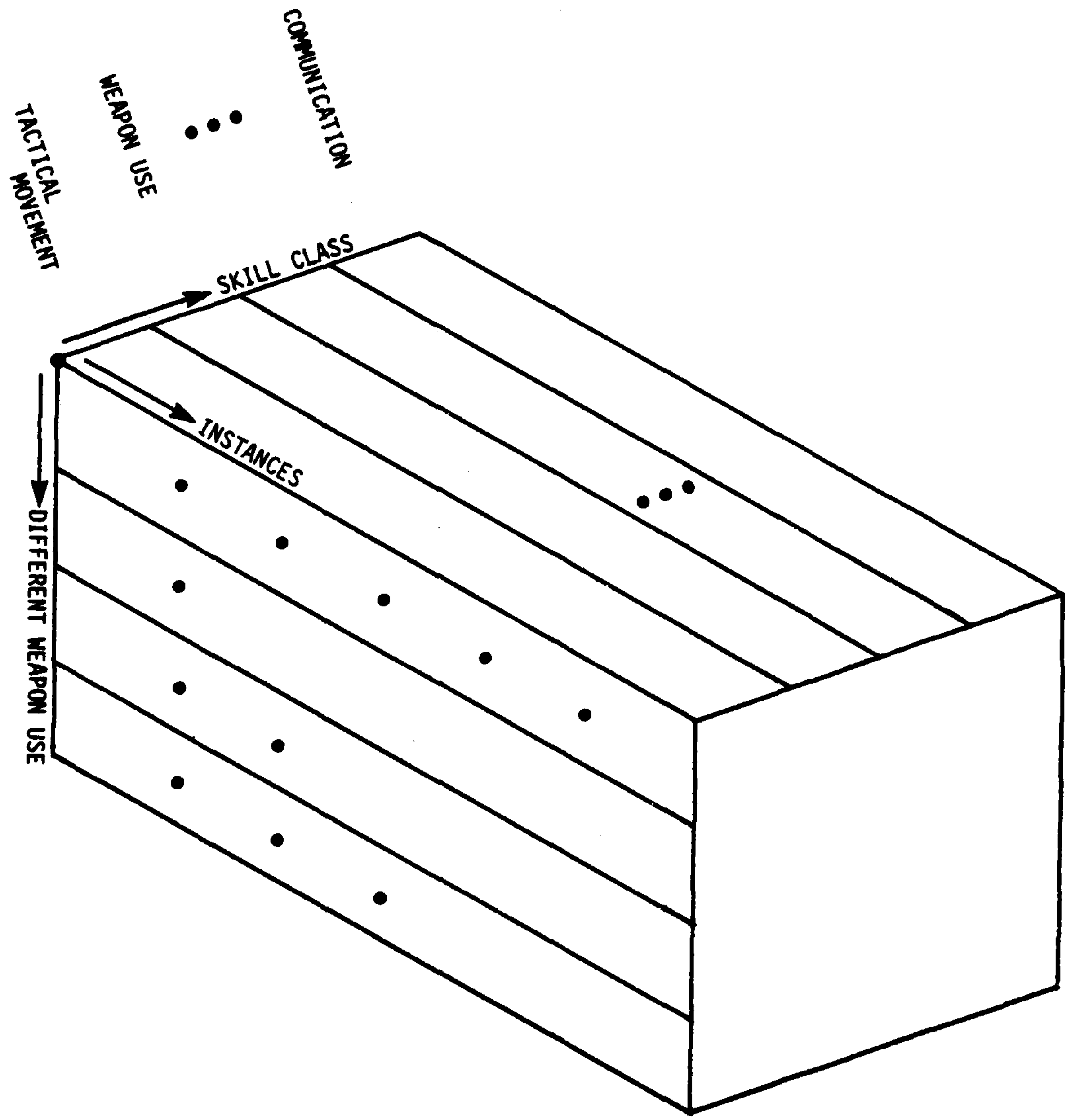

FIGURE 5-3.
POSSIBLE DIMENSIONS OF SUMMARY REPORT GENERATION

A by product of this task by task evaluation is a detailed history of the tasks performed by the training units. This history, including the specific user inputs, is used to produce the skills Evaluation Summary Report. A sample of the type of report that such a program is able to produce is given in Figure 5-4. This summary was generated from the specific scenario presented in Section 4.4. The text in square parentheses is to be picked up from the tactical knowledge base, and the specific inputs and names of tasks are given by the user. In terms of Figure 5-3, it gives a summary, instance by instance, but grouped along classes of skills. It is an easy extention to calculate an average or a total score for all instances of the same specific skill. Thus, the summary can read "the light section failed to take proper travel formation in 27% of the move to next position tasks." Higher levels of summaries can be developed to use more sophisticated parts of the knowledge base. The report in Figure 5-4 is essentially a low level summary. We will give here a few comments on each section of the report itself. Note that it can be completely generated automatically from the information collected in the interactive session and those internally stored in the knowledge base.

(1) Header. The header information is partially elicited from the user and partially derived from general information about the exercise schedule that would be available in a typical implementation of a training evaluation system.

(2) Evaluation Summary. This short summary indicates whether the top level mission was attained, the major deviations from the norm (e.g., time spent), and the major "unexpected" events that occurred; unexpected in the sense that the events and the following sequence of tasks were not the

T A N K   P L A T O O N   E V A L U A T I O N   S U M M A R Y

UNIT: Tank Div 764, Comp. 23, Platoon 96

MISSION: Daytime Hasty Attack

COMMANDER: Capt. Roger Moore

EVALUATIN OFFICER: Maj. Jim Brown

DATE OF TRAINING EXERCISE: 15-MAY-1980

DATE OF EVALUATION: 16-MAY-1980

EVALUATION SUMMARY:

The tank platoon achieved the overall mission goal of [reach assembly area]. It took [3 1/2] hours to perform the [bounding overwatch] subtas with part of the delay caused by a [sagger attack].

Number and type of casualties; [2] people lost, [3] people wounded and [1] tank lost.

TASKS PERFORMED:

The tasks performed by the platoon and its sections, shown as a hierarchy and in chronological order were as follows:

```
HASTY ATTACK
     .
     .
     .
 MOVE TO CONTACT
       .
       .
       .
   BOUNDING OVERWATCH

          A1 :  Heavy_Section [ move to Hill 204 ]
          A2 :  Light_Section [ cover Heavy_Section from Hill 1 ]
          A3 :  Heavy_Section [ take watch Hill 204 ]
          A4 :  Heavy_Section [ signal ready ],
                [ cover Light_Section from Hill 204 ]
          A5 :  Light_Section [ move to Hill 207 ]
          A6 :  Light_Section [ take watch Hill 207 ]
          A7 :  Light_Section [ signal ready ],
                [ cover Heavy_section from Hill 207 ]
          A8 :  Heavy_Section [ move to Hill 210 ]

          E1 :  [ sagger missle attack ] for [ Heavy Section ]
```

FIGURE 5-4.
SAMPLE EVALUATION SUMMARY REPORT

SAGGER ATTACK

A10: Heavy_Section [shoot back]
A11: Light_Section [shoot with all weapons]

E2 : [kill sagger] for [Light_Section]

A12: Heavy_Section [shoot]
A13: Heavy_Section resumes [move to Hill 210]

BOUNDING OVERWTCH

.
.
.

PERFORMANCE STANDARDS:

The PLATOON attained expected performance standards in [12] of [13] tasks performed during the exercise.

The unattained standards were:

. [reach Hill 210] in task [A8]

ACTION PERFORMANCE MEASURES -

The PLATOON performed [18] of the [24] prerequisite actions expected during the exercise.

The (LIGHT__SECTION) did not perform the following:

. [take proper travel formation] in task [A1]

. [take proper travel formation] in task [A8]

The [HEAVY__SECTION] did not perform the following:

. [send out an observer] in task [A2]

. [take proper travel formation] in task [A5]

. [load gun] in task [A7]

. [report enemy location] in task [A11]

TACTICAL PERFORMANCE MEASURES:

The unit demonstrated the following tactical performance levels. They are grouped under the [two] general evaluation objectives required by the training evaluator. The cases where the performance levels were unacceptable are listed individually.

1. Individual tank tactical behavior :

Out of [28] performance measures instances relevant to [individual tank tactical behavior] the evaluation results were :

a. Good in [16] cases ;
b. Acceptable in [8] cases ;

FIGURE 5-4. (CONT'D)

c. Unacceptable in the following [4] cases :
. [distances between tanks] [150] in [Heavy_Section] task [A1]
. [distances between tanks] [100] in [Heavy_Section] task [A2]
. [selection of fire area] in [Light_Section] task [A2]
. [have visual contact] in [Light_Section] task [A2]

2. Communication :

Out of [7] performance measures instances relevant to [communicatio
the evaluation results were :

a. Good in [4] cases ;
b. Acceptable in [2] cases ;
c. Unacceptable in the following [1] cases :
. [report to commanding unit] in [Heavy_Section] task [A1]

RESOURCES USED

The PLATOON used the following resources during the training exercise :

1. Casualties
Lost [2] people, out of [20]
Wounded [3] people, out of [20]

2. Main weapons
Lost [1] tank, out of [5]

3. Ammunition
Used [15] gun rounds, out of [100]
Used [2500] 0.5 gun rounds, out of [15000]

4. Fuel
Used [125] gallons, out of [750]

The resources usage are in the acceptable range .

FIGURE 5-4. (CONT'D)

normal, noneventful sequence of the mission. Casualties and major weapon use is also summarized if they exist.

(3) Task Performed. The program produces a hirarchical list (by indentation) of all the tasks and their subtasks performed by the unit and its components. This listing is used in the rest of the report as a reference for naming of tasks and events. Events that are out of the normal sequence are indicated specifically, e.g., E1, the "Saggar Missile Attack."

(4) Performance Standards Attainment. A general score and the specific standard of performance that was not attained is given.

(5) Action Performance Measures - Required Actions. This section summarizes the performance in a specific kind of performance measure-actions that have to be done at the beginning of a task.

(6) Tactical Performance Measures. Here the general scores and the cases of unacceptable levels of performance are given in several performance measure categories. These correspond to the categories indicated by the user during the initialization phase as being of interest to him in this particular evaluation. Thus, in this case, he wanted to see only an evaluation of the tank's tactical behavior and communication activity. He did not care about, e.g., command and control aspects.

(7) Resources Used. A summary of the rseources used, especially casualties and main weapons lost is given if applicable.

An important point to emphasize here is that the report is generated automatically from the user inputs by comparing them with the tactical knowledge base and aggregating to good, acceptable, and unacceptable scores for the categories of performance measures.

5.5 The Main Programs

The main program components of the evaluation aid system (TACTICS) and their main functions are listed in Figure 5-5. Figure 5-6 is a stylized version of the top level program itself. The main functional components discussed in Section 5.2 can here be clearly identified in this figure.

5.6 The Knowledge Base

The knowledge base used and manipulated by the program can be separated into thee main pieces:

(1) The relevant mission schema.

(2) The performance hierarchy.

(3) The dynamic query structures.

These three components and the main relations among them are shown schematically in Figure 5-7.

The mission schema is the computer representation of the schema relevant to the mission at hand. It contains all that was discussed in Chapter 4

TACTICS	- The top level calling program with node elicitation evaluation loop.
INITSYS	- Brings from disk and set up data structures and display.
NEEDHELP	- Asks if information is needed on system operation.
EXPSYS	- Explains the various modes of system operation.
EVALOBJ	- Asks and stores the direction of the evaluation.
IDENACT	- Identifies an activity after a transition whether expected or unexpected. Gives the proper comments in each use.
EVALPERF	- Prompt each evaluation measure, test response and give comment post the responses in the data structure. Carry through to the performance tree.
RESULTSOBT	- If relevant ask if the stated goals of activity were obtained, evaluate response and post it. Start and end location.
RESOURUSED	- If relevant ask for special resources consumed or prepared; ask on casualties and losses.
TERMINEVENT	- Identify terminating event--time, self initiated or an external event.
EVALTERM	- Evaluate if responded to the right event and if responded properly. Ask about the transition itself.
ANALYDEF	- Roll back performance measures in the performance hierarchy.
SUMMARY	- Produce verbal summary.

FIGURE 5-5.
MAIN PROGRAM COMPONENTS

```
BEGIN

     INITSYS;

           If NEEDHELP then EXPLAIN:
           TOPMENU;
           GET MISSION;
           DESCENT;  [Go down the hierarchy]
           While NOQUIT DO
                   BEGIN
                   If PLCURACT  = NIL then IDENACT (Platoon leader);
                   If CWBCURACT = NIL then IDENACT (Cub);

               If YES GOAL then EVALGOAL

               If YES ACTION then EVALACT
               If YESPERF then EVALPERF
               If YESRES EVALRES
                   TERMINATION
                   EVALTRANS
               End; [WHILE]
               If YESSUMARY then SUMMARIZE;

     END;
```

FIGURE 5-6.
PROGRAM/TOP-LEVEL ORGANIZATION

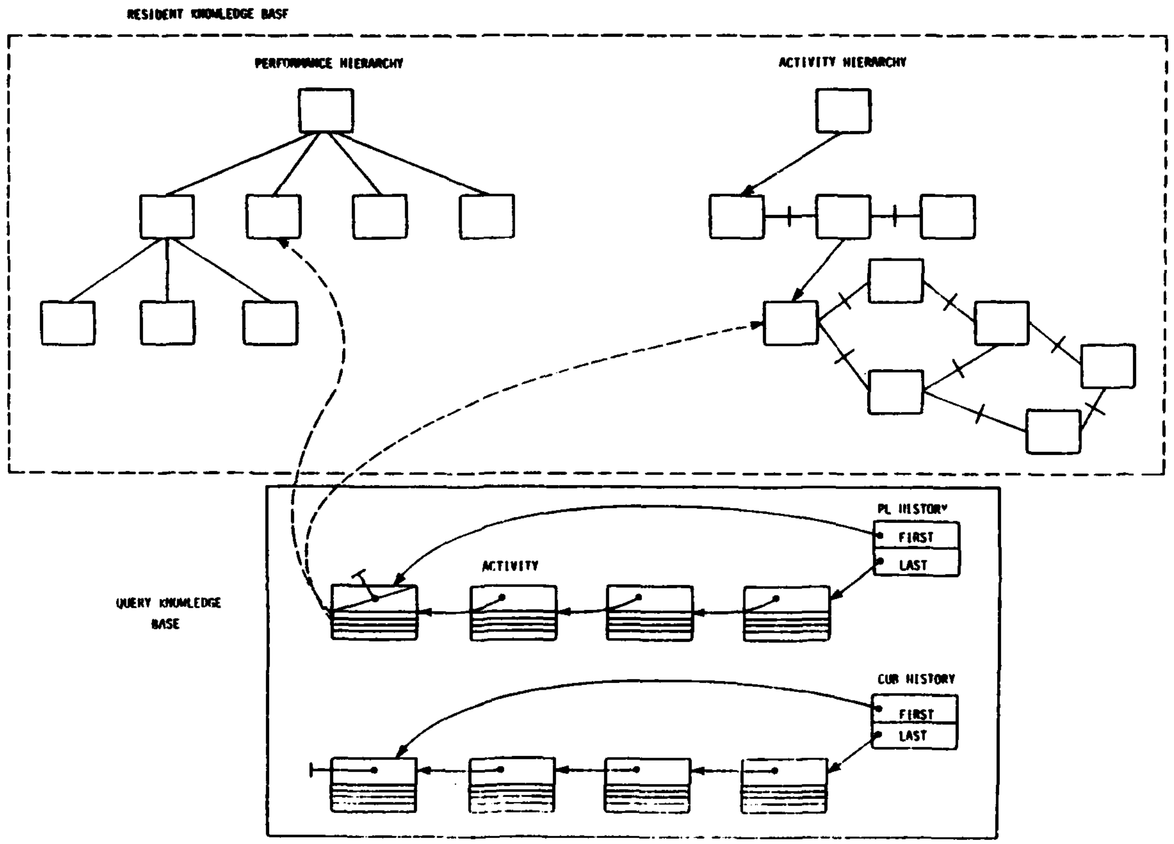

FIGURE 5-7.
KNOWLEDGE BASE HIERARCHY

about schema and more details about each performance measure, expected levels of performance, and in what way the question should be presented to the user.

The performance hierarchy (Figure 5-8) is used in the aggregation of the specific performance evaluation data elicited from the user. The user provides scores that are traced upward in in the hierarchy and accumulated scores are kept for all performance levels. In the summary, these scores are used to identify what has to be summarized and which tasks have to specifically indicated.

The query knowledge structure is the method by which the program generates a trace record of all the tasks that were performed and the levels of performance on each performance measure. It is generated dynamically as the dialog progresses. This dynamically generated history of the scenario is shown at the bottom in Figure 5-7.

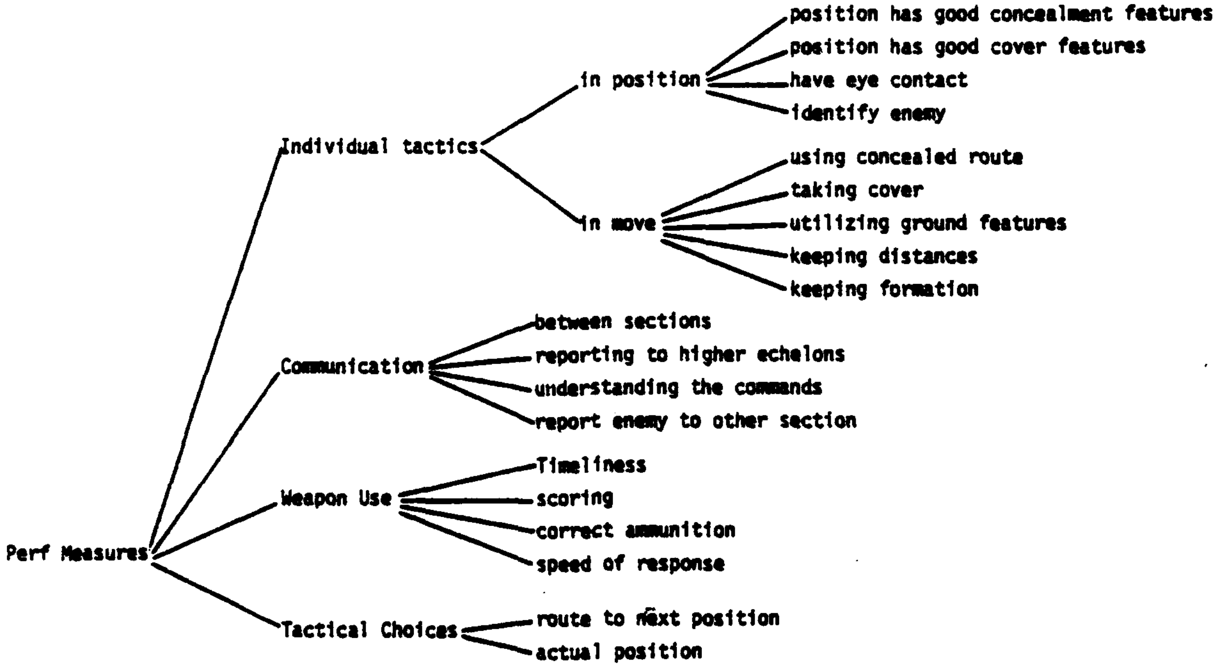

FIGURE 5-8.
A SAMPLE PERFORMANCE HIERARCHY

6. EXTENSIONS AND APPLICATIONS: DISCUSSION OF FUTURE CAPABILITIES

6.1 Overview

The effort described in this report was an exploratory study, intended as a feasibility analysis and as a demonstration of the validity of the model approach in the military training environment. This chapter will explore several directions for future extensions of the model, its application, and its implementation.

As an overall framework, we can consider future extensions along three different dimensions. These three dimensions are shown in Figure 6-1. One dimension is the military scope, which refers to expanding the scope of exercises the program can evaluate. This includes more missions for a tank platoon, larger units such as a tank company or battalion, and ultimately, combined arms exercises, where different support and defensive units cooperate in performing a mission.

The second dimension encompasses extentions in the evaluation capabilities and assistance provided by the system. As we go further along this dimension, the system assists and provides support for higher cognitive tasks of the evaluator, such as diagnosis. In this chapter, we will concentrate mainly on extentions in this direction.

The third dimension is the scope of the software implementation, which can range from a simple dedicated system, for a single user evaluating a single mission, up to very large computer systems that cover the collection of data at many locations, integration, filtering of irrelevant details, and integration with several evaluators or types of evaluation.

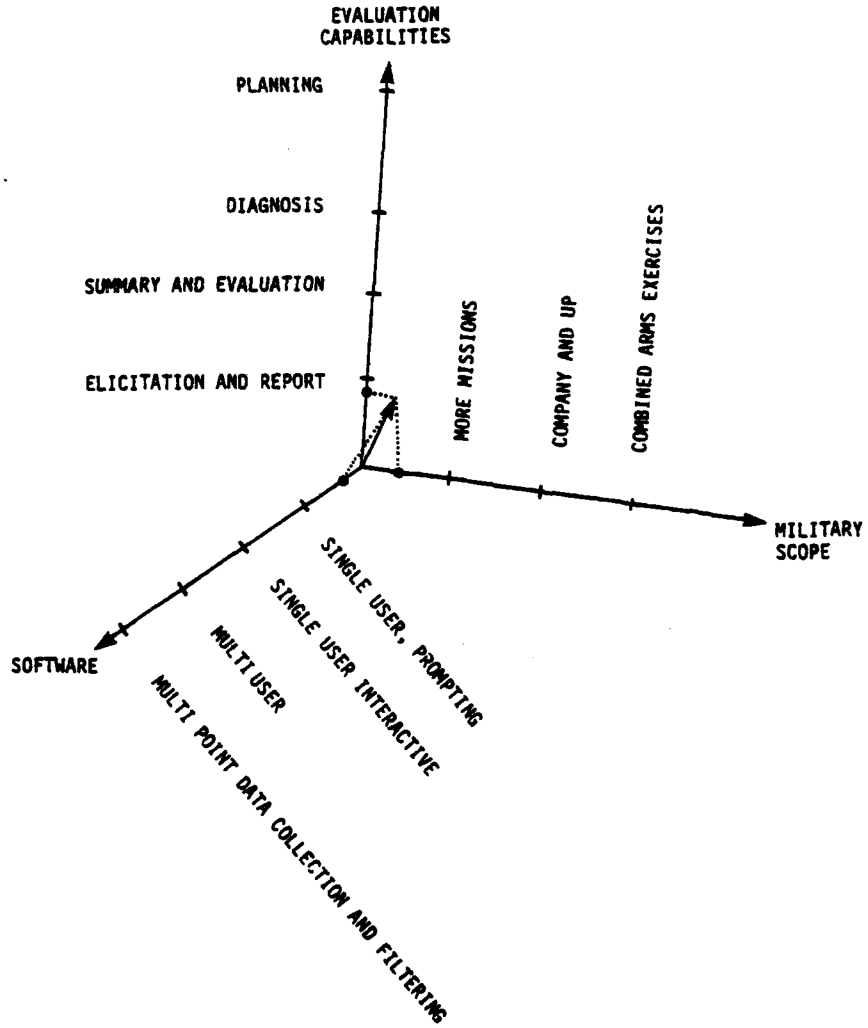

FIGURE 6-1.
DIMENSIONS OF FUTURE EXTENTIONS

6.2 Extending the Military Scope

A natural dimension to extend the application of the interactive evaluation system is to increase its military scope. Even in this dimension there are several stages in the expansion process. The first stage is to cover more of the missions of a given unit. In our demonstration program, we covered only part of a Hasty Attack of a tank platoon. This can be expanded to cover all of the Hasty Attack mission, Deliberate Attack, Area Defense, Defense of an Objective or a Route, Pursuing a Retreating Enemy or Performing an Organized Retreat. The sum of several such missions provides more assistance to an evaluator than several independent systems. This is because a tactical mission can change midstream from an offensive to a defensive task, e.g., because of an unexpectedly large OPFOR. The evaluation system then can switch easily from one type of mission to the next and can thus give a more balanced evaluation.

The second stage of expanding the military scope is expanding the size and type of units the evaluation system can accommodate. Expanding the military scope to cover the missions of a company, for example, would involve more than just a quantitative increase in the computer size. A company, and when we go higher in unit size to battalion, regiment or division, is more than a sum of its component units. There are more missions than it can be assigned, there is more variety in the maneuvers that are called for, and there is much more complexity in the roles and interactions among the subunits. The variety and complexity, however, do not surpass the capabilities of the model nor of the approach. On the contrary, they make a dynamic, interactive tool almost mandatory. As was shown in Chapters 3, 4 and 5, the model can naturally handle many concurrent interacting units performing different subtasks, where all cooperate to accomplish an overall goal. It is built around an event-

driven process that triggers the transition from one subtask to another for every independent subunit. Extention in this direction will be mainly increasing the knowledge base with more missions, more units, and more performance measures.

It appears at this stage that an evaluation aid based on our model can provide important assistance in evaluating a complex exercise such as a tank company (or battalion) performing a deliberate attack. Such an evaluation will cover all the support units, their timing communication, and the flexibility of the assistance they provide to the main offensive units. This assessment does not mean, however, that all the problems of expanding the system have been resolved. The effort would still be a research and development effort, but the objective of providing a useable tool seems well within the capability of the model and current technology.

Further expansion of the military scope along this dimension to cover combined arms missions is also conceivable. These are different from a tank battalion mission in the number and variety of the subunits involved, and in the number and variety of the missions themselves. After some experience is accumulated in applying and using the evaluation tool on small units and less complex missions, it seems feasible to apply the same methodology to larger combined arms missions. Such an expanded evaluation tool can be expected to provide the same benefits on a larger scale: i.e., increase the speed, thoroughness, and effectiveness of the evaluation process.

6.3 Expanding the Evaluation Capabilities

The second dimension along which the evaluation aid can be expanded is in the evaluation capability provided. The first level of support the

system can provide is the level provided by the demonstration program covered by this report. The system can prompt the user interactively to analyze the sequence of tasks performed by each unit, to determine the events that caused the termination of each task, and to assess the level of performance against standards obtained from the program's internal knowledge base. The system then generates a summary of the events that happened, and of areas where the unit did not meet required standards of performance.

An expanded evaluation capability would provide more sophisticated summary and evaluation mechanisms. For example, instead of indicating in which instances of "move to next position" the heavy section failed to utilize available cover and concealment, the summary process would search through all tasks that belong to the general class of "tactical moves" and then give an evaluation of the type: "The unit failed to utilize cover and concealment in 45% of the tactical moves." Such a summarization would be triggered when a specific standard of performance were not met more than, say, twice.

The specific conditions when this sort of summarization is triggered, what is an acceptable percentage of failure, and when the failure is important enough to be reported, have to be tuned through experience with a working system. In general, we want to catch all important failures, but not overwhelm the user with excessively long lists of small single-case mishaps.

Another type of evaluation that can be provided is some sort of aggregation of classes of performance measures. From many specific performance measures that fall under the same class, the program may combine the results demonstrated by the unit into a general statement: "The unit moves too slowly in offensive maneuvers." This is not an evaluation of

any specific case which may be within the acceptable range of speed, instead it is an overall judgment that the unit move maneuvers are on the low side of the range.

Various other aggregating rules should be tried and their validity assessed. For example, minimax rules are appropriate when the minimal level of unit response to a maximal enemy effort can be used to measure the readiness of the unit in a worst case situation. Maximin rules can give an estimate of the unit's readiness under best conditions, i.e., its grasp of the tactical principles of a mission. Multiattribute evaluation rules can be used to establish some global measures of performance that can be used in determining the sufficiency of training in some class of skills. It is clear that it would not be feasible to train all units, and even not one of the units, to accomplish a perfect score in all missions, so a valid aggregation method has to be devised. The knowledge base and evaluation program that are contained in the evaluation assistance tool provide the hooks to generate these aggregation and summary mechanisms. At this stage, it seems that a simple, general evaluation formula would not be satisfactory; many such aggregation rules would have to be developed; some would be triggered automatically by a peculiar (specific) feature in the data; others would be explicitly demanded by the user; and still others would be the result of both. The developent of a satisfactory set of such summarization programs is an important task in the development of an effective evaluation tool.

A third level of the expanded capabilities can be called diagnosis tools. These are a set of programs that can perform inference and deductions from the elicited set of facts about the evaluated exercise. The deductive program, starting with a set of facts observed in the field, and elicited from the user or other observers, will be able to

construct an hypothesis of a set of missing tactical skills that can account for all the inappropriate, untimely, or wrong actions made by the unit. This diagnosis capability has been demonstrated successfully in other areas, namely medical diagnosis (MYCIN and MEDAS) and geological exploration (PROSPECTOR).

Although the detailed applications are quite different, and the knowledge-base content is wide apart, the deductive processes are similar. The principle is to go from observed facts, which are probably incomplete and inaccurate, to probabilistic information about cause and effect, and come up with a small set of probable "disorders" that together manifest themselves as the set of observable facts. In the course of the deduction process, the program may ask for further information to confirm or refute a potential hypothesis. This incremental accumulation of facts is similar to that done regularly in the medical field, where more and more tests are performed to confirm or refute theories about the existing disorder, tests which would have been too expensive to perform on all patients without some a priori cause. The application of a similar technique to the military domain will also reduce the amount of data that has to be provided for the system to come up with a useful tactical diagnosis of a unit.

The amount of effort that is required to build such a training diagnosis system is not small. It will, furthermore, still be a development effort, which adds to the estimate of resources to be spent. A rough estimate on the amount of effort necessary can be obtained from the same systems mentioned above; in which it took 15-45 man-years of R&D to bring a useful diagnosis tool to the field in preliminary form. However, the benefits of these systems, even in developmental form are still tremendous. They bring the diagnostic skill and accuracy of top-level experts to the aid of the average practitioner in his everyday diagnosis

and decision making. They have the potential for a tremendous improvement in the level of medical care delivered in the field, and a similar potential for improvement in the military. The effort covered in this report is just the initial step in this direction.

6.4 Expanding the Software Scope

The software side of potential future expansions has to go hand in hand with the functional improvements. That is, there will be large expansions in terms of program and knowledge-base size and computer performance requirements associated with any expansion of the military scope and functional capability. There are also some improvements and directions of expansion that are more purely associated with the software and the computer itself, and the way they are applied. This dimension is shown in Figure 6-1 as the software dimension.

In this system aspect, we are looking at improving the interaction between the user and the program, increasing the number of users that can cooperate in a common evaluation task, and even helping in the collection of data and with some preliminary filtering.

The demonstration program elicits from the user-evaluator the facts about the exercise by prompting him point by point in a rigid sequence determined by the program. This mode of communication can be improved by making it a more free-style interaction, letting the user take the initiative in some of the issues. This can help direct the dialog toward the points that are of interest to the user at any given time.

Increasing the user's freedom in this sense, however, requires corresponding increases in the program's capability. When he changes the issue at hand, the program has to detect it and change its own so

that subsequent input will be placed in the correct intended context. The software can be organized as a set of tools that manipulates the description of the evaluated exercise being reconstructed. Instead of the program forcing the order in which the tools should be used, the responsibility and freedom is given to the user. The improved flexibility can improve acceptability by user, but increase the required training to apply the tools.

If the data that has to be collected to evaluate in detail a company or battalion exercise are too much to handle for a single user (if he cannot enter it in a reasonable length of time, or it is not available to any single person), it may be necessary to distribute the load to several evaluators and change the program accordingly. The program will have to handle several streams of facts about the exercise, to be able to resolve conflicts or contradictions, to prompt the right user for a missing key fact, etc. In all, it has to construct a single integrated description of the exercise from the information given to it by the several users. It then has to interact with the main evaluating officer and provide him with the summary or analysis.

A very rough estimate of the amount of increased effort involved in going from one user to three or four, is to multiply the programming required by a slightly larger factor. If the number of users increases substantially, e.g., 10-20, than a proper partition of the task that each performs can prevent a linear increase in the amount of effort called for.

A perhaps more likely direction of software extension is to automate the data collection. In an instrumented exercise environment, such as the future NTC, a large part of the data collection and concentration effort will be done automatically. Even in less structured engagement simula-

tion environments, such as field training centers, portable data acquisition systems can be used to record continuously the salient aspects of an exercise. Software must be provided to filter and condense the raw data coming from the collection systems. For example, when dealing with a battalion tank force, the movement of each individual tank may be unimportant, but the moves of platoons and companies are. Thus, the computer program may have to trace the moves of the "center of gravity," the main body of a platoon or a company, and this information will be used in the evaluation. Of course, casualties (disabled vehicles) will have to be counted and accumulated, but again, an overall summary number of casualties for each company will probably be sufficient. The development of what programs to write for data collection and reduction will have to wait until it is clearer what information is needed and useful for the evaluation and diagnosis system. Initial experience with such courses as the NTC should be helpful in this regard, and close examination of such experience is highly recommended as a way of proceeding in the directions outlined above.

7. REFERENCES

Agnew, J. The National Training Center: The Next Thing to the Ultimate Realism. Army, February 1980.

Buchanan, et al. Computer Assisted Chemical Reasoning. Proceedings of International Conference in Computers in Chemical Research, Education and Technology. 1976.

Freedy, A., Davis, K. B., Steeb, R., Samet, M. G. and Gardiner, P. C. Adaptive Computer Aiding in Dynamic Decision Processes: Methodology, Evaluation and Application. Perceptronics, Inc. PFTR-1016-76-8/30, August 1976.

Hart, P. and Duda, R. PROSPECTOR: A Computer Based Consultation System for Mineral Exploration, SRI Technical Note No. 155, 1977.

INTERLISP REFERENCE MANUAL. Bolt, Beranek and Newman and Xerox Corporation, 1974.

May, D., Shaket, E. and Leal, A. Knowledgeable Opponent Models for Enemy Submarine Tactics Training Simulators. Technical Report NAVTRAEQUIPCIN 78-C-0107-1, August 1979.

Nilsson, N. Problem Solving Methods in Artificial Intelligence. McGraw-Hill, 1971.

Shaket, E., Ben-Bassat, M., Madni, A., and Leal, A. Applications of Adaptive Programming Technology (APT) to Command Group Training and Performance Improvement, Perceptronics, Inc. PFTR-1055-78-12, December 1978.

Shortliffe, E. H. Computer-Based Medical Consultation: MYCIN. Elsevier, 1976.

Steeb, R. Interactive Aspects of Man-Computer Shared Decision Making, M.S. Thesis, UCLA, 1973.

APPENDIX A A SAMPLE INTERACTION WITH THE PROGRAM

Do you wish to get an explanation of the system?

Enter Y/N <RET> → Y

This is an interactive computer system for evaluation of a tactical exercise of a tank platoon. The system will ask you about the tasks that took place during the exercise, performance levels, and events that impacted its progress. After a detailed interaction

Enter <RET> to continue → <RET>

the system will generate a summary and evaluation of the exercise and the important events in it. The system is based on a dynamic, event-driven model called the PRODUCTION RULES MODEL, and was adapted to the military environment at PERCEPTRONICS, Inc.

Enter <RET> to continue ➡ <RET>

TOP LEVEL SYSTEM MENU:

1. Evaluate tactical performance.
2. Change missions, actions, or performance measures.
3. Change existing preferred solutions.
4. Define new missions or tactics.

Select one of these alternatives 1, 2, 3, or 4 <RET> ➡ 1 <RET>

SET OBJECTIVES OF THE TACTICAL EVALUATION SESSION:

1. Individual tank performance.
2. Command group selection of maneuvers.
3. Communication between sections.
4. Weapon use.

Select one of these alternatives 1, 2, 3, or 4 <RET> ➡ 1 <RET>

DIRECTIONS:

The system will now step through the tasks of the [HASTY ATTACK] mission of the tank platoon. It will ask you the events, the unit's reaction to them, and their evaluation. In particular, for each task the unit performed it will address the following:

Enter <RET> to continue ➡<RET>

These are the directions for the user. The square brackets contain an item that would have been entered by the user earlier--to select the mission to be evaluated.

1. Mission completion.
2. Performance of required actions.
3. Performance levels of specific standards.
4. Resources used.
5. The choice and transition to the next task.

Enter <RET> to continue → <RET>

```
UNIT ACTIVITIES

Heavy Section: Move to next position    Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

                                        LEVEL-3: Bounding Overwatch


        EVALUATING ATTAINMENT OF TASK OBJECTIVE:

    Did the unit achieve the objective:  [reach next
    overwatch position]?




Enter Y/N <RET>→Y
```

- *Here we jumped directly into a sample interaction where a typical task in a mission is being evaluated.*
- *At the top right the hierarchy of supertasks above the tasks under discussion are indicated, i.e., the interaction in progress is a subtask of Bounding Overwatch.*
- *At the top left we see the current actions of the platoon sections, where the heavy section is indicated to be under evaluation.*
- *The header in the middle indicates what is evaluated now.*
- *The question picked the specific data about the objective of the mission from the tactical knowledge base.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING REQUIRED INITIAL ACTIONS:

Did the unit [take correct travel formation]?

Enter Y/N <RET>➡Y

- *The required initial actions are the steps the unit has to take when beginning a task.*
- *It was also picked up from the knowledge base.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING REQUIRED INITIAL ACTIONS:

Did the unit [report the beginning of travel]?

Enter Y/N <RET>→ N

- *Another required action picked from the knowledge base and formed into a question.*
- *The answer No is stored for the later summary.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING PERFORMANCE LEVELS OF SPECIFIC STANDARDS:

What were [the distances between the tanks in motion]?

Enter answer in Yards <RET>→150

- *Now the system goes on to evaluate specific performance standards.*
- *It picks, from the knowledge base, one measure and presents it as a question in brackets [ ].*
- *This performance standard is quantitative.*

UNIT ACTIVITIES

Heavy Section: Move to next position

Light Section: Cover to heavy section

Mission: Hasty Attack

LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING PERFORMANCE LEVELS OF SPECIFIC STANDARDS:

What were [the distances between the tanks in motion]? 150

The correct distance between tanks in motion [200] to [400] [yards].

- *The computer picks the correct data from the knowledge base and presents it tutorially.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING PERFORMANCE LEVELS OF SPECIFIC STANDARDS:

Did the unit [take correct tactical route]?

Answer: Good/Acceptable/Unacceptable

Enter G/A/U <RET>→G

- *More evaluative standards, relying on the user to give assessment.*
- *This time the assessment is qualititative G/A/U.*
- *The system can handle different types of evaluative data.*

UNIT ACTIVITIES

Heavy Section: Move to next position

Light Section: Cover to heavy section

Mission: Hasty Attack

LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING RESOURCES USED:

Did the platoon have any casualties?

Enter number <RET>→3

- *Now the resources, people, and weapon systems used in the specific task are elicited and compared to expected levels.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING RESOURCES USED:

How much ammunition was used, if any?

Enter number <RET>→17

UNIT ACTIVITIES

Heavy Section: Move to next position

Light Section: Cover to heavy section

Mission: Hasty Attack

LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING TASK TERMINATION:

Did [move to next position] end by [reach next overwatch position]?

Enter Y/N <RET>→N

- *The event that terminated the Heavy Section's task move to next position is sought.*
- *The No answer will trigger the next question.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

EVALUATING TASK TERMINATION:

Did [move to next position] end by [reach next overwatch position]?

What happened?

Give name of event <RET>→Saggar Attack

- *This is an open ended question and after getting the name of the event, the system looks in its knowledge base to find a schema for the new twist in the progress of the exercise.*
- *Thus, the sequence of tasks evaluated follow the events in the actual exercise.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contack

LEVEL-3: Bounding Overwatch

IDENTIFY NEXT TASK:

What was the unit's next activity after the event [Saggar Attack]?

Enter task name, <RET>→Shoot Back

- *The system picks up the user's response and continues from there.*
- *It expects the unit to go into the first task of a response to a saggar attack.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contack

LEVEL-3: Bounding Overwatch

IDENTIFY NEXT TASK:

What was the unit's next activity after the event [Saggar Attack]?

[Shoot Back]

The expected response to: [Saggar Attack] is [take cover],

but let us consider: [shoot back] and continue.

Enter <RET> to continue ➡ <RET>

- *The computer stores the new task, presents the expected response to the saggar attack and is now ready to evaluate the task Shoot Back as it is related to Saggar Attack.*

UNIT ACTIVITIES

Heavy Section: Move to next position   Mission: Hasty Attack

Light Section: Cover to heavy section   LEVEL-2: Move to Contact

LEVEL-3: Bounding Overwatch

IDENTIFY NEXT TASK:

More tactical tasks to consider?

Enter Y/N <RET>→N

- *The system is now ready to repeat the evaluative cycle from screen (8) to (19) for the new task.*
- *It will cycle through as long as the user wishes, always picking the specific data from the knowledge base that is relevant to the task at hand.*

Do you wish to get an evaluation summary?

Indicate Y/N <RET>→Y

- *The user can get a condensed summary of the events, unit actions and its evaluation.*

The summary is printed out.

- *The summary is printed on the attached printer.*

THIS IS THE END OF THE DEMONSTRATION

APPENDIX B SOFTWARE LISTING DOCUMENT: RULE-BASED COMPUTER PROGRAM FOR EVALUATION OF COMBAT TRAINING

1. INTRODUCTION

1.1 Overview

This document contains the software documentation for a demonstration program that presents the interactive features of a program directed toward real-time aiding for automating and improving exercise evaluation in engagement simulation training systems. The document is a supplement to Perceptronics' PFTR-1070-80-7(2) entitled "Application of Rule-Based Computer Models to the Evaluation of Combat Training: A Feasibility Study." The demonstration program is founded on a rule-based, event-driven model for the representation of small-unit tasks performed by a unit and its components. The tasks are connected by production rules--conditional events that cause transitions to new tasks. This model, when contained in a computer, provides the framework for implementation of an interactive program that evaluates tactical performance during a field exercise. The internal model allows the computer program to compare directly the preferred solution of the exercise to what was actually done, and to identify the crucial intermediate steps that caused success or failure.

The program represents a typical interaction with a training officer who performs a post exercise evaluation of a tank platoon. The system tactical knowledge base is limited in scope at this stage to the "Bounding Overwatch" maneuver during the "Move to Contact" phase of a Hasty Attack. The program's responses are derived by comparing the internal description of this mission with the user's input regarding events that actually occurred during training. Assistance to the user is provided by prompting him with information from the tactical knowledge base.

The interaction with the system helps the evaluating officer get answers to the following types of questions about the exercise being evaluated:

(1) Which phases of the mission the unit failed to perform altogether.
(2) Which events (actions by OPFOR) it failed to detect, respond to, or responded inappropriately.
(3) Which procedures were not carried out appropriately.
(4) Which, if any, were the keypoints (any of the above) that contributed to mission failure.
(5) Which resources were depleted, misused or misappropriated.
(6) Which are the skills that demonstrably did not achieve acceptable levels of performance.

The program was written in the PASCAL language and implemented on a portable PDP 11/2 microcomputer system.

1.2 Required Hardware and Software Systems Support

Hardware

The computer program was developed to operate under the following hardware system configuration:

(1) Digital Equipment Corp. LSI 11/2 16 Bit Microprocessor.
(2) 64000 Bytes of RAM memory.
(3) 4 Serial I/O communication channels.
(4) Dual floppy disks with 1.2 million Bytes of backup memory.
(5) 24 x 80 full CRT display.

Software

The program uses the UCSD PASCAL (T) operating system. The program itself contains more than 1000 lines of PASCAL code and occupies about 25,000 Bytes of memory. Its reaction time for most requests by the user is not more than 2 seconds.

1.3 Organization

The software listing is divided into independent "component files" that make up the overall program. Each component file is described in a separate section with internal file nomenclature. For further reference of the listing, use UCSD PASCAL (T) software document. A sample program output and CRT display format is given in Section 3 for further reference.

2. COMPONENT FILES

```
{=====================================================================}
{                                                                     }
{         TOP  LEVEL  TEXT  FILE  OF  THE  "TACTICS"  SYSTEM          }
{                                                                     }
{              INCLUDES  ALL COMPONNENT FILES                         }
{                                                                     }
{=====================================================================}

program tactics;

{$I :TACT.G3.TEXT}                 { GLOBALS                     }
{$I :TACT.A2.TEXT}                 { INNITIALIZATION  PROCEDURES }
{$I :TACT.B3.TEXT}                 { SCREEN CONTROL              }
{$I :TACT.C3.TEXT}                 { INTERACTION PROCEDURES      }
{$I :TACT.D2.TEXT}                 { THE MAIN PROGRAM            }
```

2.1 Global Routine

```
{=========================================================================}
{                                                                         }
{ This is a demo program for the interactive evaluation of the tactical   }
{ performance of a tank platoon. The process is based on the event driven}
{ model of tactical behavior composed of activities, production rules,    }
{ and performance measures .                                              }
{                                  by EFRAIM  SHAKET                      }
{=========================================================================}


{=========== This is the globals vars and type part ==================}


const
  screenh = 24;
  screenw = 80;
  maxname = 20;
  maxdesc = 80;
  maxent  = 2 ;
  maxnode = 15;
  maxperfmeasure = 20;

type
  nametype = string[maxname];
  desctype = string[maxdesc];
  disptype = array[0..5] of string[65];
  unitype =  1..3;  {1=platoon,2=plsection,3=cubsection}
  famtype =  0..2;

  action = record
             name: nametype;
             desc: desctype;
           end;

  production = record
                 event1,event2 : desctype;
                 actname : nametype;
                 desc : desctype;
                 transto : integer ;
                 default : integer;
               end;

   resource = record
               name : nametype;
               resconsumed : integer;
               relation : string[15];   { e.g. the most, no more than }
               units : nametype ;
             end;
```

```
perfmeasr = record
               name : nametype;
               perfclass : integer ;
               evaluation : record
                               good : array[1..2] of integer;
                               accept : array[1..2] of integer;
                               units : nametype;
                            end;
            end;


perfnode = record
              name : desctype;
              level : integer;
              father,son,brother : integer;
              total,good,accept,unaccept : integer;
           end;


nodeptr = integer;
dumnode = integer;

node = record
          name: nametype;
          goal: desctype;
          level : integer;
          act : array[1..maxent] of action;
          prod : array[1..maxent] of production;
          perf : array[1..maxent] of perfmeasr;
          res  : array[1..maxent] of resource;
          who : unitype;
          father : integer;
          case sons : famtype of
            1 : ( fson,cson :integer);
            2 : ( plson,plcson,cubson,cubcson : integer );
       end;

nodeentry = record
               name :nametype;
               goal : desctype;
               diskloc : integer;
            end;

activity = record
              name : nametype ;
              node :integer ;
              glatained : integer ;  { Goal attained or not }
              perf :array[1..maxent] of integer;  {(0..2)unacc/acc/gd}
              act : array[1..maxent] of integer;  {(0..1) done/not}
              product : integer;  {production activated}
              transfer : integer; {node transfered to }
              eval : array[1..maxent] of integer; {(0..4) evaluation of tran:
           end;
```

```
var
  first,badcmd,dummy :boolean;
  ch : char;
  pftitle,nftitle,nttitle,answer,str,str2,x,prln : string;
  index,itop,jtop,curlevel : integer;

  state : record
            tasktitle : array[1..4] of string[10];
            task : array[1..4] of string[20];
            level: integer;
            plact : string[40];
            plmark : string[2];
            plhlgt,pltag : boolean;    { highlight, make visible pl action }
            cubact : string[40];
            cubmark : string[2];
            cubhlgt,cubtag : boolean;  { same for cub}
            header : string[60];
            headertag : boolean;
          end;

  status : array[1..3] of record             {platoon,pl-section,cub-section}
                           name : nametype;


                           level : integer;
                           curactivity : integer;
                           first, last : ^activity;
                          end;

  curnode,root,lastnode,nextnode,plcuract,cubcuract : ^node;
  curperf : ^perfnode;
  curactiv : ^activity;
  curact : ^action;
  curprod : ^production;
  resptr : ^resource;

  disp0,disp1,disp2,disp3,disp4,disp5,disp6 : disptype ;

  ptable :array[1..maxperfmeasure] of perfnode;
  pfile : file of perfnode ;
  pbuf : perfnode;

  ntable : array[1..maxnode] of nodeentry;
  ntabfile : file of nodeentry;
  curent : ^nodeentry;
  entbuf : nodeentry;

  nfile : file of node ;
  nbuf : node;

{-----------------------------------------------------------------------}
```

2.2 Initialization Routines

```
{------------------------------------------------------------------}
{                                                                  }
{                          :TACT.A2  file                            }
{                                                                  }
{                   THE INNITIALIZATION ROUTINES                   }
{                                                                  }
{------------------------------------------------------------------}


procedure wait(T_60 :integer);
  { Waits T_60 units of 1/60-th of a  second }

  var
    t0,t1,t2,t3,i,j:integer;

  begin
    time(t0,t1);
    t3:=t1;
    while (t3-t1)< T_60 do time(t2,t3);
    { end of wait of  }
  end;

procedure init0;

  { Sets up the innitial state display }
  var i : integer;

  begin
    with state do
      begin
        tasktitle[1]:='MISSION  :' ;          { set up the state display }
        tasktitle[2]:='LEVEL-2  :' ;
        tasktitle[3]:='LEVEL-3  :' ;
        tasktitle[4]:='LEVEL-4  :' ;
        for i:=1 to 4 do   task[i]:='                    ';
        level  := 0;
        plact  := ' ';
        pltag  := false;
        plhlgt := false;
        plmark :='  ';
        cubact  := ' ';
        cubtag  := false;
        cubhlgt := false;
        cubmark := '  ';
        header  := ' ';
        headertag := false;
      end;
  end;  {init0}
```

```
procedure initl;

  begin
    disp0[0]:='               T H E   T A C T I C S   S Y S T E M   ';
    disp0[1]:=' ';
    disp0[2]:='               FOR   TACTICAL   TRAINING   EVALUATIN      ';
    disp0[3]:=' ';
    disp0[4]:='             By: Efraim Shaket   @   PERCEPTRONICS Inc. ';
    disp0[5]:= ' ';

    displ[0]:='TOP  LEVEL SYSTEM MENU :          ';    {set up the top menue }
    displ[1]:='    1. Evaluate tactical performance   ';
    displ[2]:='    2. Change goals, actions or performance measures';
    displ[3]:='    3. Change existing prefered solutions';
    displ[4]:='    4. Define new missions or tactics';
    displ[5]:=' ';

    disp2[0]:='SET OBJECTIVES OF THE TACTICAL EVALUATION SESSION:';
    disp2[1]:='    1. Individual tank performance ';
    disp2[2]:='    2. Command group selection of maneuvers';
    disp2[3]:='    3. Communication between sections';
    disp2[4]:='    4. Weapon use  ';
    disp2[5]:= ' ';
  end;

procedure init2;

    { set up the explanation of the system }
  begin
    disp3[0]:='This is an interactive computer system for evaluation ';
    disp3[1]:='of a tactical exercise of a tank platoon. ';
    disp3[2]:='The system will ask you about the activities that took ';
    disp3[3]:='place during the exercise, performance levels, and events ';
    disp3[4]:='that impacted its progress. After a detailed interaction ';
    disp3[5]:= ' ';

    disp4[0]:='the system will generate a summary and evaluation of the ';
    disp4[1]:='exercise and the important events in it. ';
    disp4[2]:='  The system is based on a dynamic, event driven model ';
    disp4[3]:='called the PRODUCTION RULES MODEL, and was adapted to the ';
    disp4[4]:='military environment at PERCEPTRONICS Inc.              ';
    disp4[5]:=' ';
  end;
```

```
procedure init3;
  begin
    disp5[0]:='                              DIRECTIONS :    ';
    disp5[1]:='The system will step now through the activities of the ';
    disp5[2]:='[ HASTY ATTACK ] mission of a tank platoon. It will ask';
    disp5[3]:='you the events, the unit reactions to them and their ';
    disp5[4]:='evaluation. In particular, for each activity of the unit';
    disp5[5]:='it will address the folowing :    ';

    disp6[0]:='    1. Goal attainment ';
    disp6[1]:='    2. Performance of Prerequisite actions ';
    disp6[2]:='    3. Performance levels of specific measures ';
    disp6[3]:='    4. Resources used ';
    disp6[4]:='    5. The choice and transition to the next activity';
    disp6[5]:=' ';
  end;

procedure getfromdisk;
  var
    i : integer;
  begin
    reset(ntabfile,'ntfile');
    if ioresult <>0 then
      begin
        gotoxy(0,22);
        writeln('I/O error in getting node entry table file, <RET> =>');
        readln(ch);
        exit(tactics);
      end;
    for i:=1 to maxnode  do
      begin


        seek(ntabfile,i);
        get(ntabfile);
        ntable[i]:= ntabfile^ ;
      end;

    reset(pfile,'permsr');
    if ioresult <>0 then
      begin
        gotoxy(0,22);
        writeln('I/O error in getting performance measure file, <RET> =>');
        readln(ch);
        exit(tactics);
      end;
    for i:=1 to maxperfmeasure do
      begin
        seek(pfile,i);
        get(pfile);
        ptable[i]:= pfile^ ;
      end;
  end;
```

```
procedure initsys;

 {Brings knowledge base from disk and sets up the data structures and the
   display }

   begin
     init0;
     init1;
     init2;
     init3;

    with status[1] do
      begin
        name :='PLATOON';
        level:= 0;
        first := nil;
        last := nil;
      end;
    with status[2]  do        { PL section history }
      begin
        name :='PL-section';
        level := 0;
        first := nil ;
        last  := nil ;
      end;
    with status[3]  do        { CUB section history }
      begin
        name :='CUB-section';
        level:= 0;
        first := nil ;
        last  := nil ;
      end;
     { A dummy initialization }

     new(curnode);
     root:= curnode;

     with curnode^  do
       begin
         name:= 'Move to next pos';
         goal:= 'reach next overwatch position';
         father:=0 ;
         who:= 2 ;   { PL }
         level:= 3;
       end;
```

```
  curlevel:=curnode^.level;
  with state do
    begin
      level:=curlevel ;
      task[1]:=' Hasty Attack ';
      task[2]:=' Move To Contact';
      task[3]:=' Bounding Overwatch';
      task[4]:=' ';
    end;
  gotoxy(0,0);
  write('Initializing .');
  for index:=1 to 8 do
    begin
      wait(60);
      write('.');
    end;
  writeln(' ');
end;
```

2.3 Screen Control and Display Procedures

```
{-------------------------------------------------------------------}
{                                                                   }
{                         :TACT.B3   FILE                            }
{                                                                   }
{            SCREEN  CONTROL  and  DISPLAY PROCEDURES               }
{                                                                   }
{-------------------------------------------------------------------}


procedure screen(code:integer);

{ sends control commands to a Hazeltine CRT }

      { clear screen              28          }
      { clear to end of line      15          }
      { clear to end of screen    24          }
      { home cursor               18          }
var

  trans : packed array [1..2] of char;
{I-}
begin
  trans[1] := chr(126);              { Lead in char to screen }
  trans[2] := chr(code);             { Insert code of command }
  unitwrite(1,trans,2)
end;
{I+}


procedure dispstate;

{ Displays system state in the different fields on the screen, each according
{ to a tag associated with it. All the information is in the STATE record.

  var  i : integer;
  begin
    screen(28) ;   { clear the screen }
    with state do
      begin
        gotoxy(10,0);
        writeln('UNITS ACTIVITIES ');
        if plhlgt then  begin
                          plmark := '=>';
                          cubmark:= '  ';
                        end
                  else  begin
                          plmark := '  ';
                          cubmark:= '=>';
                        end;
```

```
      if pltag then
        begin
          gotoxy(2,2);
          screen(15);
          writeln(plmark,'PL-SECTION :  ',plact);
        end;
      if cubtag then
        begin
          gotoxy(2,3);
          screen(15);

          writeln(cubmark,'CUB-SECTION:  ',cubact);
        end;
    for i:=1 to 4  do
      begin
        gotoxy(50,i);
        screen (15);      { clear to end of line }
      end;
    for i:=1 to level do
      begin
        gotoxy(50,i);
        write(tasktitle[i],task[i]);
      end;
    if headertag then
      begin
        gotoxy(10,8);
        screen(15);
        writeln(header);
      end;
    screen(18);              { home cursor }
  end;
end;

procedure showscr(col,row,inc,num: integer; text : disptype);
  { display a whole screeful at location row,col }
  var
    i:integer;
  begin

    for i:=0 to (num-1)  do
      begin
        gotoxy(col,(row+i*inc));
        writeln(text[i])
      end;
  end; {showscr}

  procedure prompt ;
   {  Prompt for a <RET> and get a return character }
   var  c : string;

   begin
     gotoxy(0,22);
     write('Enter <RETURN> to continue  =>');
     readln(c)
   end;
```

```
procedure getans(col,row:integer; str:string; var c : char );

{ Show a line, prompt  and get a one character answer }

begin

  gotoxy(col,row);
  write(str,' <RETURN>  => ');
  readln (c)   ;
end;
```

2.4 Actual User Interaction Procedures

```
{=========================================================================}
{                                                                         }
{                              :TACT.C3   FILE                            }
{                                                                         }
{                  THE ACTUAL USER INTERACTION PROCEDURES                 }
{                                                                         }
{=========================================================================}


procedure start;

{ Displays the opening screen }


  begin
    screen(28);
    showscr(5,8,2,6,disp0);                        {the opening screen}
    prompt;
    wait(60*2);
  end;

  procedure getyesno(var cc : char);
    { Get a yes or no answer }

    begin
      repeat
        gotoxy(0,22);
        screen(15);
        getans(0,22,'Enter  Y / N  ',cc);
        badcmd:= not(cc  in ['y','n']);
        wait(45);
      until not badcmd;
    end;


function  needhelp : boolean;
 { If general information about the system's operation is needed, a short
   description is given here. }

  begin
      screen(28);
      needhelp:=false;
      gotoxy(5,12);
      writeln('Do you wish to get an explanation of the system?');
      getyesno(ch);
      case ch of
        'y' :  needhelp := true;
        'n' :  needhelp := false;
      end;
      wait(3*60);
  end;
```

```
procedure explain;
  { A short explanation of the system }
  var
    nothing : integer;  {a dummy}

  begin
    screen(28);
    showscr(5,8,2,6,disp3);
    prompt;
    screen(28);
    wait(1*60);
    showscr(5,8,2,6,disp4);
    prompt;
    wait(2*60);

  end;


procedure conskwb;
    begin end;

procedure topmenu ;
  { Display top menue snd get answer }
  var
    answer : string;

  begin
    screen(28);
    showscr(5,8,2,6,disp1);
    repeat
      gotoxy(0,22);
      screen(15);
      getans(0,22,'Select one of these alternatives 1 2 3 or 4',ch);
      badcmd := not (ch in ['1','2','3','4']);
      if not badcmd then
        begin
          case ch of
            '1' :  badcmd:= false;
            '2','3','4' : begin
                            gotoxy(0,22);
                            screen(15);
                            writeln('This alternative is not implemented ');
                            wait(2*60);
                            badcmd := true;
                          end;
          end;
        end;
      wait(30);
    until not badcmd;

    wait(3*60);

  end;
```

```
procedure evalobject;
  { Ask about evaluation objectives }

  var
    dummy : integer;
    answer : string;

  begin
    screen(28);
    showscr(5,8,2,6,disp2);
    repeat
      gotoxy(0,22);
      screen(15);
      getans(0,22,'Select one of these alternatives 1 2 3 or 4 ',ch);
      badcmd := not (ch in ['1','2','3','4']);
      if not badcmd then
        begin


        case ch of
          '1' :  badcmd:= false;
          '2','3','4' : begin
                          gotoxy(0,22);
                          screen(15);
                          writeln('This alternative is not implemented ');
                          wait(2*60);
                          badcmd := true ;
                        end;
        end
      end;
      wait(45);
    until not badcmd;
    wait(60);
  end;


procedure get_mission ;
  begin  end;


procedure directions ;
  begin
    screen(28);
    showscr(5,8,2,6,disp5);
    prompt;
    screen(28);
    wait(1*60);
    showscr(5,8,2,6,disp6);
    prompt;
    wait(2*60);
  end;
```

```
function noquit :boolean ;
  { Asks if user wants to quit }

  begin
    if not first then
       begin
         dispstate;
         gotoxy(5,12);
         screen(15);
         writeln('More tactical activities to consider  ?');
         getyesno(ch);
         case ch of
           'y' :  noquit  := true;
           'n' :  noquit  := false
         end;
         first:=false;
         wait(2*60);
       end
       else
         begin
            noquit := true;
            first:=false;
         end;
  end;


procedure idenact(inunit : unitype);
  begin end;


function yesgoal : boolean;

    { Checks if the current node has a specific goal }

   begin
     yesgoal:=true;
   end;

 procedure evalgoal(nptr : dumnode);
   begin
     with state do
       begin
         header := 'EVALUATING GOAL ATTAINMENT:';
         headertag := true;
       end;
     dispstate;
     gotoxy(5,12);
     writeln('Did the unit achieve the goal : [',curnode^.goal,'] ?');
     getyesno(ch);

     wait(3*60);
   end;
```

```
procedure evalperf(nptr : dumnode);
  begin

    with state do
      begin
        header := 'EVALUATING PERFORMANCE MEASURES :';
        headertag := true;
      end;
    dispstate;
    wait(60);

    gotoxy(5,12);
    writeln('What were [ the distances between the tanks in motion ] ?');
    wait(60);
    getans(0,22,'Enter answer in YARDS    ',ch);

    gotoxy(5,15);
    writeln('The proper distances are [ 200 ] to [ 400 ] YARDS    ');

    wait(2*60);
    gotoxy(5,12);
    screen(24);
    writeln('Did the unit [ take correct tactical route ]  ? ');
    wait(60);
    gotoxy(5,15);
    writeln('Answer Good/Acceptable/Unacceptable  ');
    wait(60);
    repeat
      gotoxy(0,22);
      screen(15);
      getans(0,22,'Enter G / A / U ',ch);
      badcmd:= not (ch in [ 'g','a','u']);
      wait(45);
    until not badcmd;
    wait(3*60);
  end;


function yesres : boolean;

   { Cheks for comments on resources usage  }

  begin
    yesres:=true ;
  end;
```

```
procedure evalres(nptr : dumnode);
  begin
    with state do
      begin
        header := 'EVALUATING RESOURCES  USED :';
        headertag := true;
      end;
    dispstate;
    wait(60);

    gotoxy(5,12);
    writeln('Did the platoon have any casualties  ? ');
    wait(90);
    getans(0,22,'Enter number  ',ch);
    wait(2*60);

    gotoxy(5,12);
    screen(24);
    writeln('How much ammunition was used, if any    ? ');
    wait(45);
    getans(0,22,'Enter  number ',ch);

     wait(3*60);
   end;

 procedure termination (nptr : dumnode);
    { Identifies the terminating event for the current activity }

   begin
     with state do
       begin
         header := 'EVALUATING ACTIVITY TERMINATION :';
         headertag := true;
       end;
     dispstate;
     wait(60);

     gotoxy(5,12);
     writeln('Did [ ',curnode^.name,' ]  end by [ ',curnode^.goal,' ]  ? ');
     getyesno(ch);
     case ch of
       'n' : begin
               wait(1*60);
               gotoxy(5,15);
               writeln('What happened  ? ');
               wait(90);
               gotoxy(0,22);
               write('Give name of event   <RET>  =>');
               readln(str);
               gotoxy(5,15);
               screen(24);
               writeln(str);
               wait(2*60);
             end;
       'y' : begin
               str := 'reached hill 204';
             end;
     end; {case}
   end;
```

```
procedure evaltransition ;
  { Identifies the next activity even if it is unexpected, then evaluates }
  { the transition itself .                                               }

  begin
    with state do
      begin
        header := 'IDENTIFY NEXT ACTIVITY  :';
        headertag := true;
      end;
    dispstate;
    wait(90);
    gotoxy(5,12);
    writeln('What was the activity after [ ',str,' ] ? ');
    wait(30);

    gotoxy(0,22);
    write('Enter the activity name, <RET> => ');
    readln(str2);
    state.plact := str2;
    wait(90);

    gotoxy(5,15);
    screen(24);
    writeln(str2);
    screen(24);
    gotoxy(5,18);
    writeln('The expected response to : [ ',str,' ] is [ take cover ], ');
    gotoxy(5,20);
    writeln('but let us consider : [ ',str2,' ]  and continue.  ');
    prompt;
    wait(3*60);

  end;

  {$G+}

procedure summarise ;
  { presents a summary from two disk files }

  label 1;
  var
    sum1,sum2 : interactive;
    i,j,k  :integer;
    s : string;
```

```
begin
  screen(28);
  gotoxy(5,12);
  writeln('Do you wish to get an evaluation summary ?');
  repeat
    gotoxy(0,22);
    screen(15);
    getans(0,22,'Indicate  1 / 2   ',ch);
    wait(4*60);
    badcmd:= not(ch in ['1','2']);
    case ch of
      '1' :  s := ':suml.text' ;
      '2' :  s := ':suml.text' ;
    end;
  until not badcmd;
  {$I-}

  reset(suml,s );
  repeat
    gotoxy(0,0);
    screen(28);
    for i:=0 to 20 do
      begin
          if eof(suml) then goto 1;
          readln(suml,s);
          writeln(s);
      end;
    repeat
      gotoxy(0,22);
      screen(15);
      getans(0,22,'Continue ? Y / N ',ch);
      badcmd := not (ch in [ 'y','n']);
      if ch = 'n' then goto 1 ;
    until not badcmd;
  until eof(suml);
1:
  {$I+}
  prompt;
  wait(3*60);
  close(suml,lock);
end;
```

2.5 The Main Procedure of the Tactics System

```
{=========================================================================}
{                                                                         }
{                           :TACT.D2    FILE                              }
{                                                                         }
{              THE  MAIN  PROCEDURE  OF THE TACTICS SYSTEM                }
{                                                                         }
{=========================================================================}

Begin                        { Beginning of main program }

  initsys ;
  if needhelp then explain ;
  topmenu ;
  evalobject;
  get_mission ;
  directions ;
  first:=true;
  while noquit do
    begin
      with state do
        begin
          plact := 'move to contact';
          pltag := true;
          cubact := 'cover PL section';
          cubtag := true;
          header := 'STARTING THE ELICITATION :';
          headertag := true;
          plhlgt := true;
          level := 3;
        end;
      dispstate;
      wait(3*60);

      if yesgoal then evalgoal(index  ) ;

      if yesaction then evalaction(index  ) ;

      if yesperf  then evalperf(index  ) ;

      if yesres  then evalres(index  ) ;

      termination( index) ;

      evaltransition ;

    end { while };

  summarise;
  screen(28);
  gotoxy(5,10);
  writeln (' THIS IS THE END OF THE DEMO ');
  prompt;
end {system}.
```

3. PROGRAM DISPLAY AND SAMPLE OUTPUT

3.1 Interaction Display

Figure 3-1 illustrates the CRT display as it is seen by the user during the dialog. The dashed lines are imaginary borders that outline where each type of information is displayed. The principles of consistent and informative man-machine dialogs were strictly maintained. Each area of the screen presents the same kind of information at all times. At the top right corner, the mission hierarchy is given so that the user always knows the mission and the hierarchy of tasks under it down to one level above the task he is communicating with. This provides a global view. At the top left are the specific tasks "currently" (the time in the exercise) under discussion. The subunit considered at the moment is highlighted with an arrow. At the center of the screen the type of evaluation currently being done is presented. The rest of the lower part of the screen is used for the free format dialog with a prompting line at the bottom indicating the specific kind of answer expected.

3.2 A Sample Evaluation Summary Report

The output of the evaluation program is an Evaluation Summary Report. A sample of the type of report that such a program is able to produce is given in Figure 3-2. We will give here a few comments on each section of the report:

(1) Header. The header information is partially elicited from the user and partially derived from general information about the exercise schedule that would be available in a typical implementation of a training evaluation system.

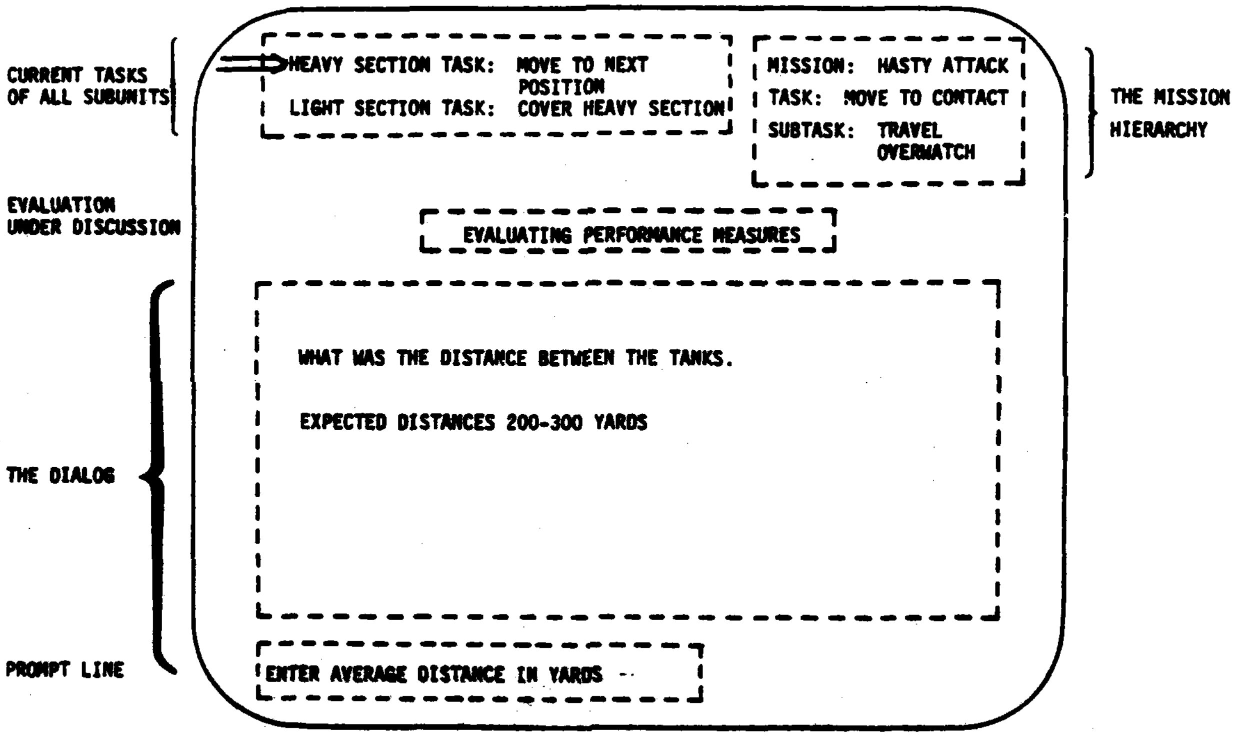

FIGURE 3-1.
A SAMPLE INTERACTION DISPLAY

T A N K   P L A T O O N   E V A L U A T I O N   S U M M A R Y

UNIT: Tank Div 764, Comp. 23, Platoon 96

MISSION: Daytime Hasty Attack

COMMANDER: Capt. Roger Moore

EVALUATIN OFFICER: Maj. Jim Brown

DATE OF TRAINING EXERCISE: 15-MAY-1980

DATE OF EVALUATIN: 16-MAY-1980

EVALUATION SUMMARY:

The tank platoon achieved the overall mission goal of [reach assembly area]. It took [3 1/2] hours to perform the [bounding overwatch] subtas with part of the delay caused by a [sagger attack] unexpected activity.

Number and type of casualties; [2] people lost, [3] people wounded and [1] tank lost.

TASKS PERFORMED:

The tasks performed by the platoon and its sections, shown as a hierarchy and in chronological order were as follows:

```
HASTY ATTACK
      .
      .
      .
  MOVE TO CONTACT
        .
        .
        .
    BOUNDING OVERWATCH

        A1 :  Heavy_Section [ move to Hill 204 ]
        A2 :  Light_Section [ cover Heavy_Section from Hill 1 ]
        A3 :  Heavy_Section [ take watch Hill 204 ]
        A4 :  Heavy_Section [ signal ready ],
              [ cover Light_Section from Hill 204 ]
        A5 :  Light_Section [ move to Hill 207 ]
        A6 :  Light_Section [ take watch Hill 207 ]
        A7 :  Light_Section [ signal ready ].
              [ cover Heavy_section from Hil. 207 ]
        A8 :  Heavy_Section [ move to Hill 210 ]

        E1 :  [ sagger missle attack ] for [ Heavy Section ]
```

FIGURE 3-2.
SAMPLE EVALUATION SUMMARY REPORT

SAGGER ATTACK

A10: Heavy_Section [shoot back]
A11: Light_Section [shoot with all weapons]

E2 : [kill sagger] for [Light_Section]

A12: Heavy_Section [shoot]
A13: Heavy_Section resumes [move to Hill 210] .

BOUNDING OVERWTCH

.
.
.

PERFORMANCE STANDARDS ATTAINMENT:

The PLATOON attained expected performance standards in [12] of [13] tasks performed during the exercise.

The unattained standards were:

. [reach Hill 210] in task [A8]

PERFORMANCE MEASURES - DOING REQUIRED ACTIONS:

The PLATOON performed [18] of the [24] prerequisite actions expected during the exercise.

The (LIGHT_SECTION) did not perform the following:

. [take proper travel formation] in task [A1]

. [take proper travel formation] in task [A8]

The [CUB_SECTION] did not perform the following:

. [send out an observer] in task [A2]

. [take proper travel formation] in task [A5]

. [load gun] in task [A7]

. [report enemy location] in task [A11]

TACTICAL PERFORMANCE MEASURES:

The unit demonstrated the following tactical performance levels. They are grouped under the [two] general evaluation objectives required by the training evaluator. The cases where the performance levels were unacceptable are listed individually.

1. Individual tank tactical behavior :

Out of [28] performance measures instances relevant to [individual tank tactical behavior] the evaluation results were :

a. Good in [16] cases ;
b. Acceptable in [8] cases ;

FIGURE 3-2. (CONT'D)

c. Unacceptable in the following [4] cases :
. [distances between tanks] [150] in [Heavy_Section] task [A1
. [distances between tanks] [100] in [Heavy_Section] task [A8
. [selection of fire area] in [Light_Section] task [A2]
. [have visual contact] in [Light_Section] task [A2]

2. Communication :

Out of [7] performance measures instances relevant to [communicatio
the evaluation results were :

a. Good in [4] cases ;
b. Acceptable in [2] cases ;
c. Unacceptable in the following [1] cases :
. [report to commanding unit] in [Heavy_Section] task [A1

RESOURCES USED

The PLATOON used the following resources during the training exercise :

1. Casualties
Lost [2] people, out of [20]
Wounded [3] people, out of [20]

2. Main weapons
Lost [1] tank, out of [5]

3. Ammunition
Used [15] gun rounds, out of [100]
Used [2500] 0.5 gun rounds, out of [15000]

4. Fuel
Used [125] galons, out of [750]

The resources usage are in the acceptable range .

FIGURE 3-2. (CONT'D)

(2) Evaluation Summary. This short summary indicates whether the top level mission was attained, the major deviations from the norm (e.g., time spent) and the major "unexpected" events that occurred, unexpected in the sense that the events and the following sequence of tasks are not the normal, noneventful sequence of the mission. Casualties and major weapon use is also summarized if they exist.

(3) Task Performed. The program produces a hierarchical list (by indentation) of all the tasks and their subtasks performed by the unit and its components. This listing is used in the rest of the report as a reference for naming of tasks and events. Events that are out of the normal sequence are indicated specifically, e.g., E1 the "Saggar Missile Attack."

(4) Performance Standards Attainment. A general score and the specific standard of performance that was not attained is given.

(5) Action Performance Measures - Required Actions. This section summarizes the performance in a specific kind of performance measure, actions that have to be done at the beginning of a task.

(6) Tactical Performance Measures. Here the general scores and the cases of unacceptable levels of performance are given in several performance measure categories. These correspond to the categories indicated by the user during the initialization phase as being of interest to him in this particular evaluation. Thus in this case, he wanted to see only an evaluation of the tank's tactical behavior and communication activity. He did not care about, e.g., command and control aspects.

(7) Resources Used. A summary of the resources used, especially casualties and main weapons lost is given if applicable.

An important point to emphasize here is that the report is generated automatically from the user inputs by comparing them with the tactical knowledge base and aggregating to good, acceptable, and unacceptable scores for the categories of performance measures.